CARE FOR CANE

PROJECT REPORT

21AD1513- INNOVATION PRACTICES LAB

Submitted by

SWETHA B	**(211422243327)**
SARADHA SHREE R	**(211422243287)**
SWETHA T	**(211422243331)**

in partial fulfillment of the requirements for the award of degree

of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123

ANNA UNIVERSITY: CHENNAI-600 025

November, 2024

BONAFIDE CERTIFICATE

Certified that this project report titled "**CARE FOR CANE**" is the bonafide work of **SWETHA B (211422243327), SARADHA SHREE R (211422243287), SWETHA T (211422243331)** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

INTERNAL GUIDE
Mrs.K.TAMILSELVI M.E.,
Assistant Professor,
Department of AI & DS.

HEAD OF THE DEPARTMENT
Dr.S.MALATHI M.E., Ph.D
Professor and Head,
Department of AI & DS.

Certified that the candidate was examined in the Viva-Voce Examination held on ……

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Agriculture is considered as the foundation of the Indian economy, which owns the highest of 17% of GDP. Out of all the crops grown, Sugarcane has the highest production value among all other commercial crops. As per the researches, 15%of sugarcane leaves are found to be defective, which drastically lowers both the quantity and quality of production. Digitization has a noticeable impact on severalkey areas, including harvesting plants, monitoring plant growth and smart cultivation techniques. Plant diseases, however, continue to pose a serious threatto the world's food security and have a declining effect on agricultural yields. To lessen the negative effects of the diseases, it is crucial to identify them and apply control measures. Controlling these losses can be greatly aided by early detectionof plant diseases. Farmers can contact digital centers and seek assistance from plant pathologists and experts to obtain the appropriate remedies by using the pool of digital information. Deep learning techniques have become more widely applicable in recent years. Systems for diagnosing plant diseases have been discussed for a while and are regarded as a standard technique for identifying andcategorizing plant diseases. With its ability to extract features and learn efficiently, deep learning has made significant strides in the classification of images. It has been a trend to apply deep learning based convolutional neural networks to computer vision task. Fast crop damage mitigation may be possible with the help of the proposed architecture, which may facilitate the early diagnosis and detection of plant diseases. The proposed system CARE FOR CANE (CFC) – A web platform is intended to detect early stages of leaf diseases in sugarcane and it developed by using the Rexnet-150 model. It offers promising solution for farmers to increase their sugarcane productivity.

Keywords : Sugarcane,CFC,ReXNet,Agriculture,Leaf disease,CNN,Detect

ACKNOWLEDGEMENT

I also take this opportunity to thank all the Faculty and Non-Teaching Staff Members of Department of Computer Science and Engineering for their constant support. Finally I thank each and every one who helped me to complete this project. At the outset we would like to express our gratitude to our beloved respected Chairman, **Dr.Jeppiaar M.A.,Ph.D,** Our beloved correspondent and Secretary **Mr.P.Chinnadurai M.A., M.Phil., Ph.D.,** and our esteemed director for their support.

We would like to express thanks to our Principal, **Dr.K.Mani M.E., Ph.D.,** for having extended his guidance and cooperation.

We would also like to thank our Head of the Department, **Dr.S.Malathi M,E.,Ph.D.,** of Artificial Intelligence and Data Science for her encouragement.

Personally we thank **Mrs.K.Tamilselvi M.E.**, Assistant Professor in Department of Artificial Intelligence and Data Science for the persistent motivation and support for this project, who at all times was the mentor of germination of the project from a small idea.

We express our thanks to the project coordinators **Dr.A.Joshi M.E., Ph.D.,** Professor & **Dr.S.Chakaravarthi M.E.,Ph.D.,** Professor in Department of Artificial Intelligence and Data Science for their Valuable suggestions from time to time at every stage of our project.

Finally, we would like to take this opportunity to thank our family members, friends, and well-wishers who have helped us for the successful completion of our project.

We also take the opportunity to thank all faculty and non-teaching staff members in our department for their timely guidance in completing our project.

SWETHA B

SARADHA SHREE R

SWETHA T

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS		MEANING
CNN	–	Convolutional Neural Network
REXNET	–	Rank Expansion Network
HTML	–	Hyper Text Markup Language
CSS	–	Cascading Style Sheet
JS	–	JavaScript
ReLU	–	Rectified Linear Unit
GDP	–	Gross Domestic Product

CHAPTER 1

INTRODUCTION

1.1 SUGARCANE LEAF DISEASE DETECTION

Sugarcane is the crop with the highest production value among all those grown for commercial use. Sugarcane leaf disease is a significant challenge for the farmers, causing crop destruction and financial losses. Early detection and treatment of these disease is crucial for preventing further damage, but farmers may lack the expertise to identify them. Therefore for knowing the accurate disease which affects the sugarcane leaves, deep learning techniques can be utilised to recognize the disease of the plant leaf. In order to detect the disease earlier we proposed a sustainable system which is CARE FOR CANE -A web platform which offers solution to farmers. By leveraging advancements in digital technology and deep learning, CARE FOR CANE (CFC) aims to empower farmers with an accessible, AI-powered tool that enables real-time disease identification. Through a web platform utilizing the Rexnet-150 deep learning model, the project facilitates efficient and reliable identification of disease symptoms from images of sugarcane leaves. Through CARE FOR CANE, farmers can proactively manage crop health by detecting diseases at early stages, consulting with agricultural experts, and receiving targeted treatment recommendations.

1.2 DEEP LEARNING TECHNIQUES

Deep learning is a class of machine learning algorithms that uses multiple layersto progressively extract higher- level features from the raw input. For example, in image processing, while lower layers may identify edges, while higher layers may identify the concepts relevant to a human such as digits, letters, or faces. Theword “deep” in “deep learning” refers to the numbers of layers throughwhich thedata is transformed. Deep learning uses “neural networks”

which are inspired bythe human brain. The more layers, the “deeper” the network, allowing it to learnmore complex features and perform more sophisticated tasks. It results in a widevariety of applications (computer vision, speech recognition, object detection, etc) .

1.2.1 CONVOLUTIONAL NEURAL NETWORK (CNN)

The term CNN refers to the type of ANN. CNN are now widely used for image classification, image segmentation, face recognition and object recognition. CNNcan receive any data input, including images, video, sound, speech and natural language. However CNN is simply a stack of several layers, beginning with a convolution layer and progressing through the following layers: Pooling, Relu Activation and ending with a Fully Connected layer. As a result, each image received as input will be filtered, reduced and corrected several times, to finally form a vector.

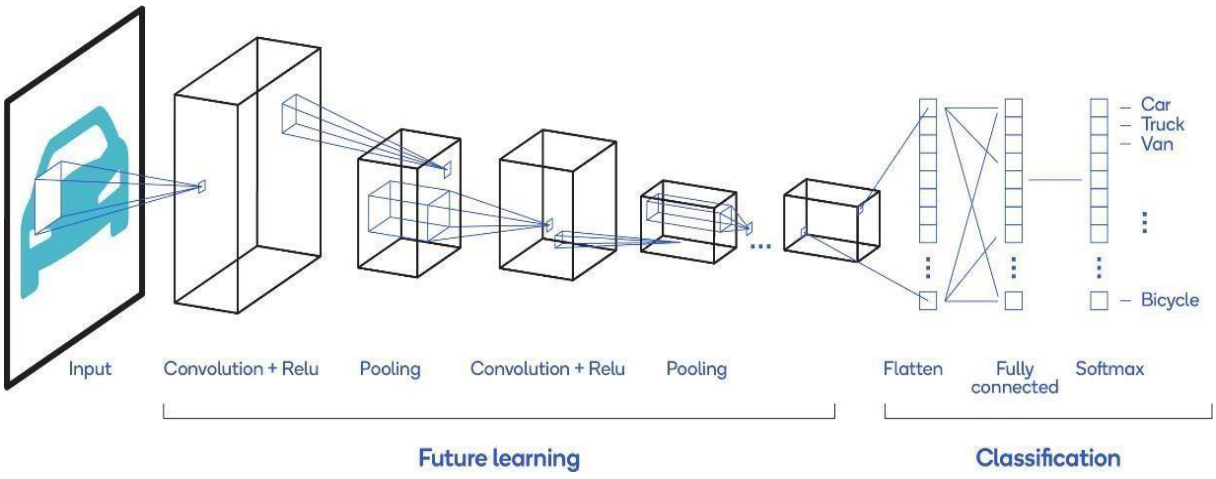


Fig1.1 CNN Architecture

1.2.1.1 INPUT LAYERS

It’s the layer in which we give input (image or a sequence of images to our model.

1.2.1.2 CONVOLUTIONAL LAYERS

The layer used to extract feature from the input dataset. It applies a set of learnable filters known as the kernel to the input images. The filters/Kernels are smaller matrices usually 2x2, 3x3, or 5x5 , it slides over the input image data.

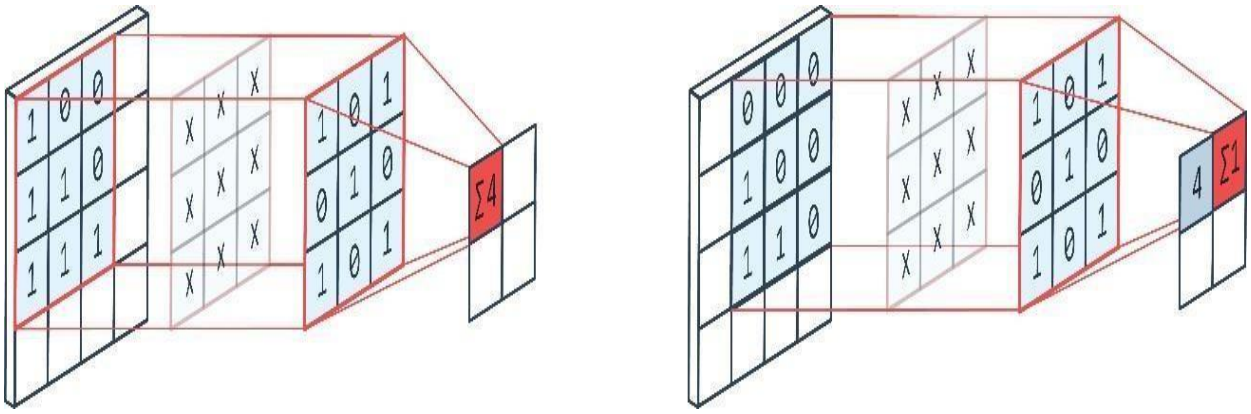


Fig.1.1.1 Convolution layer

1.2.1.3 ACTIVATION LAYERS

By adding an activation function to the output of the preceding layer, activation layer add non linearity to the network. Relu(0,x) . Relu activation layer

were used which consider negative values from input as zero and positive value as it as.

1.2.1.4 POOLING LAYERS

This layer is periodically inserted and its main function is to reduce the size of volume which makes the computation fast reduces memory and also prevents overfitting. Two common types of pooling layers are Max pooling and Average pooling.

4	6	10	12
7	8	3	7
11	14	19	2
23	17	1	5

FIG 1.1.2 POOLING LAYER

1.2.1.5 MAXIMUM POOLING LAYER:

8	12
23	19

FIG 1.1.3 MAX POOLING LAYER

1.2.1.6 AVERAGE POOLING LAYER:

5	10
20	15

FIG.1.1.4 AVERAGE POOLING

1.2.1.7 FLATTENING

The resulting feature maps are flattened into a one – dimensional vector after the convolutional layers and pooling layers.

1.2.1.8 FULLY CONNECTED LAYERS

It takes the input from the previous layer and computes the final classification or regression task.

1.2.1.9 OUTPUT LAYER

The output from the fully connected layers is then fed into logistic function for classification tasks like softmax which converts the output of each class into probability score of each class.

1.2.2 RANK EXPANSION NETWORK-150 (ReXNet-150)

RexNet-150 plays a central role as the deep learning model responsible for detecting and classifying leaf diseases in sugarcane plants. RexNet-150, a convolutional neural network (CNN) architecture optimized for efficient feature extraction and accurate image classification, is specifically chosen for this project due to its ability to handle complex visual data while maintaining high accuracy

and computational efficiency**.** Rank Expansion Networks (ReXNet) follow a set of new design principles for designing bottle necks in image classification models. It refines each layer by:

Expanding the input channel size of the convolution layer and

Replacing the ReLUs.

CHAPTER-2

2.LITERATURE SURVEY

A scholarly , which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Literature reviews are secondary sources, and do not report new or original experimental work. Most often associated with academic-oriented literature, such reviews are found in academic journals, and are not to be confused with book reviews that may also appear in the same publication. Literature reviews are a basis for research in nearly every academic field. A narrow-scope literature review may be included as part of a peer-reviewed journal article presenting new research, serving to situate the current study within the body of the relevant literature and toprovide context for the reader. In such a case, the review usually precedes the methodology and results sections of the work.

2.1 SUGARCANE LEAF DISEASE DETECTION USING A HIDDEN MARKOV MODEL

For detecting the disease they had used a hidden Markov model and anisotropic diffusion algorithm . Using the system user is able to detect the disease in 2 approach - select symptoms by comparing sugarcane (static approach) and select image from gallery or capture an image directly using a camera (dynamic approach).It helps to remove noise from the digital image without blurring edges. Hence, anisotropic diffusion is an iterative procedure where a moderately basic arrangement of calculations are utilized to figure each progressive picture in the family and this procedure proceeds until an adequate level of smoothing is obtained. The Hidden Markov Model is used for image segmentation. It works pixel by pixel for selecting the pixel from the digital image for processing. Afterselecting the necessary part that is diseased part, we mask the green part of a leaf.It results in an enhanced image of a leaf (image 1). A concealed Markov model can be viewed as speculation of a blended display where the shrouded factors (orinactive factors), which control the blended segment to be chosen for every perception, are connected through a Markov procedure as opposed to free of one another. It reduces the

preprocessing compared to other algorithms of image classification. The prior knowledge and human effort in design is a major advantage for the classification of the images. A convolutional neural system comprises of information and a yield layer, just as various shrouded layers. The concealed layers of a CNN normally comprise of convolutional layers, RELU layer, for example, initiation work, pooling layers, completely associated layers and standardization layers. Depiction of the procedure as a convolution in neuralsystems is by tradition. Scientifically it is a cross-relationship as opposed to a convolution. They have studied different paper on disease detection and they decided to use the Hidden Markov Model, Anisotropic Diffusion Algorithm andConvolutional Neural Network.

AUTHOR: Snehal Pawar, Jyotismita Talukdar

YEAR: 2023

2.2 SUGARCANE DISEASE DETECTION USING MOBILENETV2 MODEL

Sugarcane is a recurrent crop, it is cultivated on a large scale in various states of India like Maharashtra, Uttar Pradesh, Tamil Nadu, Karnataka, Bihar, and many other states. Natural disasters such as floods and storms are the primary reasons for a damaging crop in that agricultural field and viruses or bacteria are the secondary reasons that infect a plant. It also decreases the quality due to infectiousdiseases. To maintain the quality of crops, diseases control is highly need Generally,diseasesin crops are recognized by such farmers who have vast experience in farming also some agricultural scientists are helping them with disease identification. Sometimes due to changes in weather, it is difficult to identify variations in disease. It makes it difficult in identifying diseases in sugarcane. For detecting diseases, they have used the Mobile Net v2 model as of now, which is majorly used in object detection. Firstly a dataset is collected whichhas been categorized into four categories namely ret rot, wheat rust, yellow leaf,and eye

spot. Further, it is being classified into 80:20 datasets for training and testing purposes. In this process different algorithm is used to train the model andcheck the accuracy of the respected model and then compare it with other algorithms. In this, we have used two algorithms CNN, Mobilenetv2 which is a transfer learning algorithm. After the training, the file is saved using the tflite filewhich is TensorFlow Lite . It is a kind of Machine learning with the Google framework which can be used to deploy various models on a variety of devices that have platforms like Desktop, Android, and iOS. After saving, tflite model isfed into an android studio that uses Kotlin language A general-purpose, statically typed, cross-platform programming language with a type interface is entitled Kotlin. Google announced that Kotlin is now their recommended programming language for developers who develop Android apps. Instead of the default Java compiler, Kotlin has been accessible since the release of Android Studio 3.0 in October2017. After programming in kotlin, it will detect and predict the system output and will give the respected remedies. They offer experimental strategies in their work to boost accuracy by up to 90.

AUTHOR: Vaishali Wadhe, Rashmi Dongre, Yash Kankriya

YEAR: 2023

2.3 SUGARCANE DISEASE CLASSIFICATION USING DEEP LEARNING ALGORITHMS

The identification of crop diseases is one of the major concerns that the agricultural industry has to deal with. The detection and classification of leaves is essential in agriculture, forestry, rural medicine, and other commercial applications, among other things. The diagnosis of sugar cane plant leaf disease is required for automatic weed identification in precision agriculture. This paper discusses a novel approach to the development of a plant disease recognition model that is based on sugar cane leaf image classification and employs deep convolutional networks to recognise disease in sugar cane plants. The method used

for identification and automatic recognition investigates the possibility of using k-NN and SVM in pre-training with ANN, followed by CNN-based approaches for recognition. By using the KNN attained accuracy at 75 percent, SVM attained the accuracy at 80 percent, ANN attained accuracy at 61 percent and CNN attained the accuracy at 88 percent. By comparing these algorithms paper has been thoroughly trained by Convolutional Neural Network (DCNN).The structure utilised to categorise the sugar cane leaf using a simple convolutionary neural community with 6 unique instructions, the accuracy achieved is 88 percent.

AUTHOR: LAKSMIKANTH PALETI, KISHORE KUMAR et.al

YEAR: 2023

2.4 INTELLIGENT DISEASE DETECTION IN SUGARCANE PLANTS

Sugarcane is an important crop, but its production is hampered by problems such as water shortages and disease. This article presents a machine learning-based approach to accurately detect and classify sugarcane leaf diseases using convolutional neural networks (CNNs), random forests, and support vector machine models. This study focuses on the global importance of sugarcane, the prevalence of sugarcane in medical applications, and the main cultivation regions. The implementation includes the use of Python programming and deep learning algorithms, specifically his SVM, random forest, and CNN to classify sugarcane leaf diseases based on color, texture, and shape features. This process includes data collection, local binary patterns, texture analysis using Gabor filters and his GLCM, and disease classification.

AUTHOR: Usha Rani, M., Saravana Selvam, N. and Jegatha Deborah, L.

YEAR: 2022

2.5 ENHANCING SUGARCANE DISEASE CLASSIFICATION WITH ENSEMBLE DEEP LEARNING

Deep Learning practices in the agriculture sector can address many challenges faced by the farmers such as disease detection, yield estimation, soil profile estimation, etc. In this paper, disease classification for the sugarcane plant and the experimentation involved thereby is thoroughly discussed. Experimental results include the performances of the well-known existing transfer learning techniques and proposed ensemble deep learning based architecture that incorporates stack ensemble of two networks with one having level-wise spatial attention helping to provide better generalization. A Self-created database of sugarcane leaf diseases is introduced to the research community through this paper. It involves 5 categories with a total of 2569 images. Here, it is observed that best performing transfer learning method, MobileNet-V2 shows an accuracy of around 84% with the lowest number of parameters whereas ensemble model reaching to 86.53% with less epochs and with acceptable number of parameters.

AUTHOR: Swapnil Dadabhau Daphal and Sanjay M. Koli

YEAR: 2023

2.6 SUGARCANE LEAF DISEASE DETECTION USING TRANSFER LEARNING

This paper discusses in detail about using DenseNet201, a transfer learning model, along with Support Vector Machine in the output layer to detect Red Rot and Red Rust diseases in sugarcane leaves. Agriculture is crucial to the Indian economy as it provides employment to roughly half of India's population and contributes to 17% of India's GDP. Since 1947, India has seen an enormous increase in the yield and produce of crops. Yet around 50,000 crore worth of crops are lost to pest and disease attacks every year. According to the United Nations Food and Agriculture organization, there is an approximate loss of 40% in production of crops globally

due to pests and diseases. This costs the global economy more than $220 billion annually. One of the most significant cash crops grown by farmers in India is Sugarcane. Red rot and Red rust epidemics have been common for sugarcane cultivators in India. With the rise in technology and artificial intelligence, there are various methods that can provide a solution to this issue. Our paper discusses in detail about using DenseNet201, a transfer learning model, along with Support Vector Machine in the output layer to detect Red Rot and Red Rust diseases in sugarcane leaves.

AUTHOR: Shilpa Lambor ,Vithika Pungliya, Roshita Bhonsle,Atharva Purohit,Ankur Raut,Aayushi Patel

YEAR: 2022

2.7 PREDICTIVE ANALYSIS OF SUGARCANE PATHOGENS: AMACHINE LEARNING APPROACH

Sugarcane cultivation is vital for the global sugar industry, but it is threatened by various pathogens that can cause significant yield losses. Predictive analysis techniques, especially machine learning, offer a promising approach to mitigating the impact of these pathogens on sugarcane production. In this study, we propose a machine learning-based predictive analysis framework for identifying and predicting the occurrence of sugarcane pathogens. The framework involves collecting and analyzing large datasets of environmental variables, disease incidence rates, and other relevant factors to develop accurate predictive models. Various machine learning algorithms are employed to train these models and predict the likelihood of pathogen outbreaks. Through the integration of advanced analytics techniques, such as feature selection and ensemble learning, our framework aims to improve the accuracy and reliability of predictions. The results demonstrate the potential of machine learning in enhancing the management of sugarcane pathogens, enabling timely interventions and increasing the overall

resilience of sugarcane crops to disease outbreaks. This research contributes to the growing body of knowledge on precision agriculture and demonstrates the capability of machine learning in addressing complex agricultural challenge.

AUTHOR: Ananya Kamble,Shraddha Kenjale ,Srivaramangai R

YEAR: 2024

2.8 DETECTION OF DISEASES IN SUGARCANE USING IMAGE PROCESSING TECHNIQUES

India is one of the largest producers of sugarcane and ranks second in the world. The cropping season and duration of sugarcane depends on the varieties and sowing time. The time taken for its maturity is between 12 and 18 months. With high sensitivity to the environment, it easily gets affected by numerous diseases and pests. If the affected plant is not identified and taken adequate measures at the right time, it will affect the whole yield. The present study focuses on detecting various diseases in sugarcane leaves using the image processing techniques and developing a web application for the farmers to detect the major diseases of sugarcane as well. The system collects the images of the leaves and processes by means of Adaptive Histogram Equalization (AHE) superseded by segmentation using k means clustering algorithm. Using Gray Level Co-occurrence Matrix (GLCM) and Principal Component Analysis (PCA), statistical characteristics such as variance, skewness, standard deviation, mean, and covariance are extracted. Finally, the detection and classification are implemented using Support Vector Machine (SV M) that results the average accuracy value of 95%. If any variety disease is identified,the required control measures are also suggested. Fungal diseases such as brown spot, wilt, rust, red rot, and smut caused by photo plasma and viral diseases like sugarcane streak mosaic, sugarcane mosaic virus, yellow leaf syndrome affect the

yield greatly. Pests such as sugarcane borer, white wooly aphid are also not the least responsibility for the reduced yield of sugar productivity. Being a prominent cash crop of India, it tops the list by serving feed for live stocks, as fuel and its stubble and roots as organic manure. It is reported to have more than fifty diseases in sugarcane leaves.

AUTHOR: Al Hiary et al

YEAR: 2011

2.9 SUGARCANE LEAF DISEASE DETECTION AND SEVERITY ESTIMATION BASED ON SEGMENTED SPOTS IMAGE

About 15% of sugarcane leaf is defective because of diseases, it reduces the quantity and quality of sugarcane production significantly. Early detection and estimation of plant disease is a way to control these diseases and minimize the severe infection. This paper proposes a model to identify the severity of certain spot disease which appear on leaves based on segmented spot. The segmented spot is obtained by thresholding a* component of L*a*b* color space. Diseases spots are extracted with maximum standard deviation of segmented spot that use for detection the typeof disease using classification techniques. The classifier is a Support VectorMachine (SVM) which uses L*a*b* color space for its color features and Gray Level Co-Occurrence Matrix (GLCM) as its texture features. This proposed modelcapable to determine the types of spot diseases with accuracyof 80% and 5.73 error severity estimation average.

AUTHOR: Ratnasari,Evy Kamilah ; Mentari, Mustika ; Dewi, Ratih Kartika et al

YEAR: 2019

2.10 SUGARCANE DISEASES DETECTION USING OPTIMIZED CONVOLUTIONAL NEURAL NETWORK WITH ENHANCED ENVIRONMENTAL ADAPTATION METHOD

This research aims to address the need for accurate and prompt identification of sugarcane diseases, which substantially impact the worldwide sugar industry and the livelihoods of numerous farmers. Conventional visual inspection methods are hindered by subjective interpretations and restricted availability, prompting the investigation of more sophisticated techniques. By harnessing deep learning capabilities, specifically Convolutional Neural Networks (CNNs), and further enhancing their performance using the Environmental Adaptation Method (EAM) optimization, this research demonstrates significant enhancements in disease detection accuracy, precision, recall, and F1-Score. Based on the macro values obtained from the different approaches, it has been observed that an accuracy of 89% was obtained for the CNN designed from EEAM in comparison to the other counter parts. Similarly, the precision of the proposed architecture of CNN is better in comparison to GA, PSO and DE. On the same lines the Recall and F1 score of the proposed approach is better in comparison to that of the three counterparts. Similarly, the ROC analysis for the analysis of AUC is done and it was identified that the AUC curve for the different CNN designed by various optimizer were good in identifying the different classes of the sugarcane diseases. The major limitation of this approach is that model has marginal accuracy with its counterpart algorithm, however, the algorithm suggested the use of simple CNN models that are easy to use. The rigorous methodology, encompassing data collection and model optimization, guarantees the reliability and applicability of the sugarcane disease detection system based on Convolutional Neural Networks (CNN).

Future research directions focus on integrating hyperspectral imaging, unmanned aerial vehicles (UAVs), and user-friendly mobile applications. This integration aims to empower farmers, facilitate proactive disease management, and ensure the sustainability of the sugarcane industry. This advancement represents notable progress agriculture and disease mitigation.

AUTHOR: Upadhye, S. A., Dhanvijay, M. R., & Patil, S. M
YEAR: 2023

CHAPTER – 3

SYSTEM ANALYSIS

EXISTING SYSTEM

The existing systems for sugarcane leaf disease detection using deep learning involve various approaches and techniques. These systems aim to enhance disease classification and detection in sugarcane plants. One approach is the utilization of a convolutional neural network (CNN) trained as an image classifier with around 3000 leaf images. This method achieves specific accuracy in detecting and classifying sugarcane leaf diseases. The trained model can predict the class of the disease infected by the sugarcane plant, with the predicted class displayed on a mobile screen for user convenience.Another study trained and tested a deep learning model with a dataset of 13,842 sugarcane images, achieving a high moderate accuracy for disease-infected leaves and healthy leaves. This model demonstrates the effectiveness of deep learning in detecting and classifying sugarcane leaf diseases. Some of the surveys on sugarcane leaf disease identification using deep learning highlights the potential of automated image capturing systems for deep learning-based disease detection and recognition. This approach can significantly improve the efficiency and accuracy of disease detection in sugarcane crops. A sugarcane leaf dataset, consisting of 6748 images, is available for researchers to develop and evaluate models using deep learning, featureextraction and pattern recognition. This dataset can contribute to the development of more accurate and efficient disease detection systems for sugarcane leaves.These existing systems for sugarcane leaf disease detection using deep learning showcase the potential of deep learning models, automated image capturing

systems and large datasets to improve disease classification and detection in sugarcane plants using many of the deep learning models such as DenseNet, MobileNet, Residual Network. These advancements can contribute to early disease identification and improved agricultural practices.

DISADVANTAGES OF EXISTING SYSTEM

There are several disadvantages present in the existing techniques of sugarcane leaf disease detection models. Some of them are :

- Moderated Accuracy Level
- High Space complexity network models
- Limited Automation in Deep Learning Models
- High Cost of Multispectral and Hyperspectral Cameras.

PROPOSED SYSTEM

The traditional methods of disease identification in plants relies on specialist's visual observations, which can be challenging to implement on large farms due to the need for a significant workforce and continuous supervision. In many countries, farmers lack access to experts and resources, making professional consultation expensive and time-consuming. To address this issue, automated disease detection based on leaf indications offers a more cost-effective and convenient solution for monitoring large crop fields. While visual disease prognosis can be time-consuming, less accurate, and limited to specific areas, an automated detection model would enhance accuracy, efficiency and reduce the required time and effort. Common sugarcane ailments include bacterial infections, yellow leaf disease, wheat rot, viral infections and red

cotton leaf disease. Imageprocessing techniques, such as image segmentation, play a crucial role in evaluating diseased areas and identifying color variations in affected regions, aiding in the classification and analysis of plant diseases.The proposed system for sugarcane leaf disease detection using the RexNet-150 model involves particular components to achieve the higher accuracy of disease detection. The system requires images of sugarcane leaves as input. These images can be captured using a camera or a mobile device. The acquired images are preprocessed to remove noise, normalize brightness and contrast and resize the images to a standard size. The RexNet-150model is trained using a dataset of sugarcane leaf images. It comprises ofmore than 150 layers and it replaces the ReLU(Rectified Linear Unit) which acts as Activation layer. The model is trained to classify the images into different classes based on the type of disease present in the leaf. The trained model is tested using a separate dataset of sugarcane leaf images.The model predicts the class of the disease infected by the sugarcane plant. The predicted class is displayed as the result. The user can use this information to take appropriate action to manage the disease.

ADVANTAGES OF PROPOSED SYSTEM

2.10.1 The RexNet-150 model is a deep learning model that has been shown achievehigh accuracy in image classification tasks.

2.10.2 The system is designed to be user-friendly, with a simple interface that allowsusers to easily capture and classify sugarcane leaf images.

2.10.3 The system is cost-effective, as it does not require expensive equipment orspecialized training to use.

CHAPTER - 4

SYSTEM REQUIREMENTS

4.1 HARDWARE REQUIREMENTS

- Processor : AMD Ryzen 3 3250U, 2.650 GHz
- RAM : 8 GB
- Hard Disk : 915 GB
- Keyboard : Standard Keyboard
- Monitor : 15-inch color monitor

4.2 SOFTWARE REQUIREMENTS

- Operating System : Windows 11
- Front End : HTML, CSS, JS, BOOTSTRAP
- Back End : PYTHON, FLASK, GOOGLE COLAB

CHAPTER-5

SYSTEM DESIGN

5.1 SYSTEM ARCHITECTURE

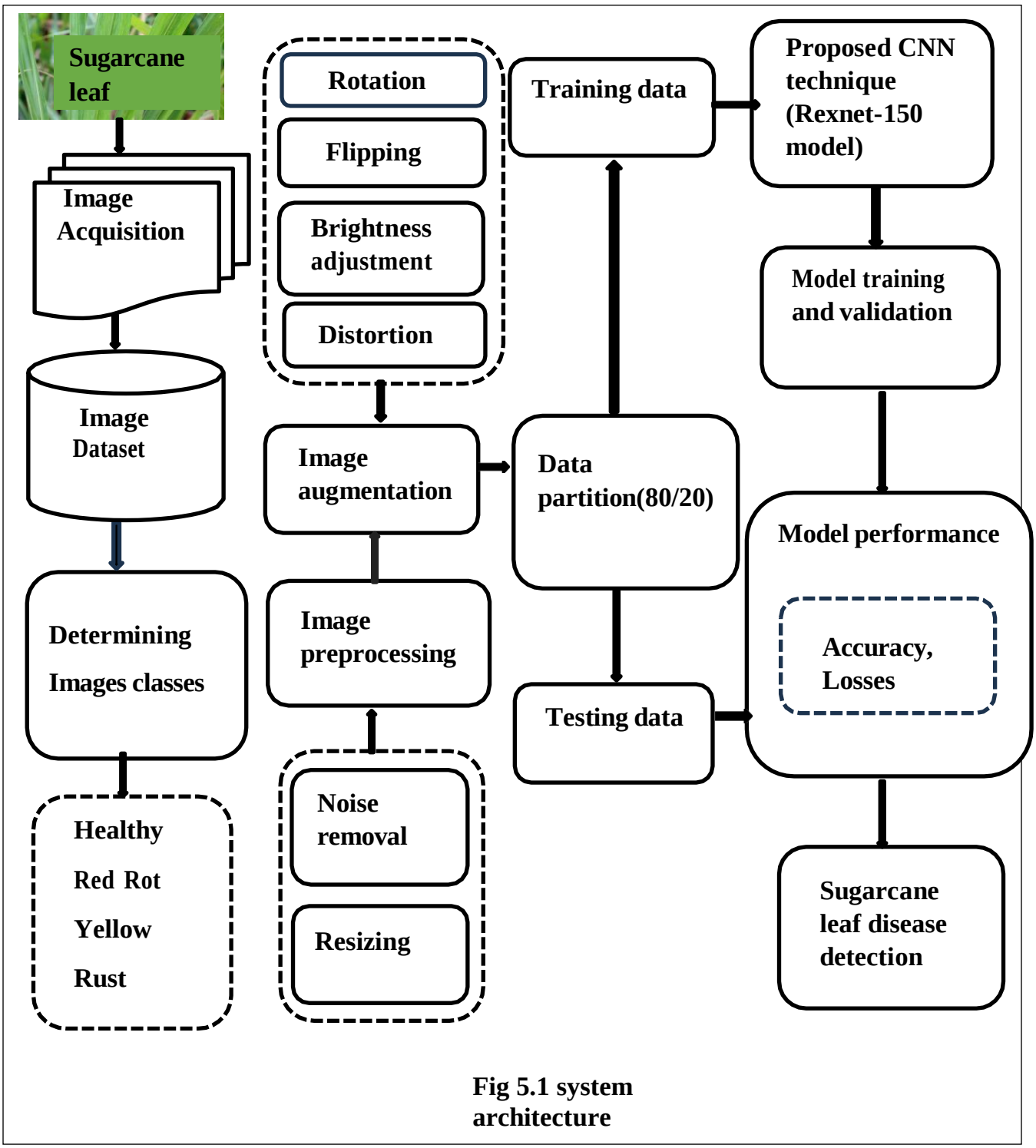


Fig 5.1 system architecture

CHAPTER-6
MODULE DESCRIPTION

LIST OF MODULES

- Image Identification
- Preprocessing
- Feature Extraction
- Image Classification
- Disease Detection

6.1 IMAGE IDENTIFICATION

Image identification is the process of identifying a feature in a image. It is a class of deep neural networks that are used to analyze visual imagery. Image identification allows to identify and discern the nature of an image or its components.

6.2 PREPROCESSING

Preprocessing refers to a set of image processing techniques applied to enhance the quality of acquired images and prepare them for subsequent analysis. This may include tasks like noise reduction, resizing, and normalization to ensure consistency and clarity across images.

6.3 FEATURE EXTRACTION

Convolutional Neural Networks are today's building blocks for image classification tasks using machine learning. However, another very useful task they perform before

classification is to extract relevant features from an image. Feature extraction is the wayCNNs recognize key patterns of an image in order to classify it. Among the many existing pretrained architectures that we can useas feature extractors, we can mention the RexNet- 150.Rexnet-150 is an architecture that incorporates the concept of bottleneck. This architecture is highly used in problems of feature extraction in computer vision.

6.4 IMAGE CLASSIFICATION

Image classification involves assigning labels or classes to input images. It is a supervised learning task where a model is trained on a labeled image to predict the class of unseen images. CNN are commonly used for image classification as they can learn hierarchical features like edges, textures, and shapes. CNN excel in this task because they can automatically extract meaningfulspatial features from images. Here are different layers involved in the process like input layer, convolutional layer, pooling layer, fully connected layer, output layer.

6.5 DISEASE DETECTION

This is the core component responsible for identifying whether a sugarcane leaf is infected with a disease or not. You can use a variety of techniques to perform disease detection. Popular deep learning based approaches using Convolutional Neural Network (CNN) such as RexNet-150 automatically learn to detect diseases within the images using 150 layers. The algorithm is trained on a dataset containing labeled images of diseased and healthy sugarcane leaves**.**

CHAPTER – 7

SYSTEM IMPLEMENTATION

7.1 METHODOLOGY

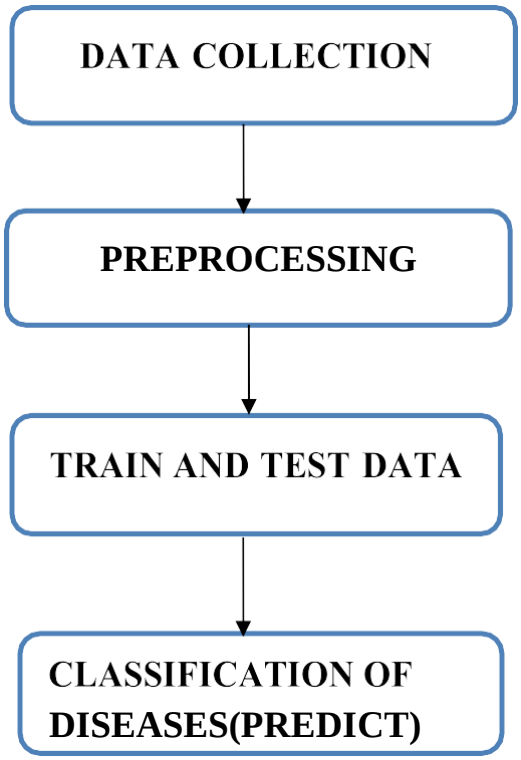


Fig 7.1 Methodology

7.1.1 DATA COLLECTION

In this project image dataset were used which had 2531 images and these images were used for training and testing. The dataset consist of five classes of images.The classes consist of Healthy, Redrot, Rust, Yellow, Mosaic leaves of sugarcane.80:20 rule is applied.

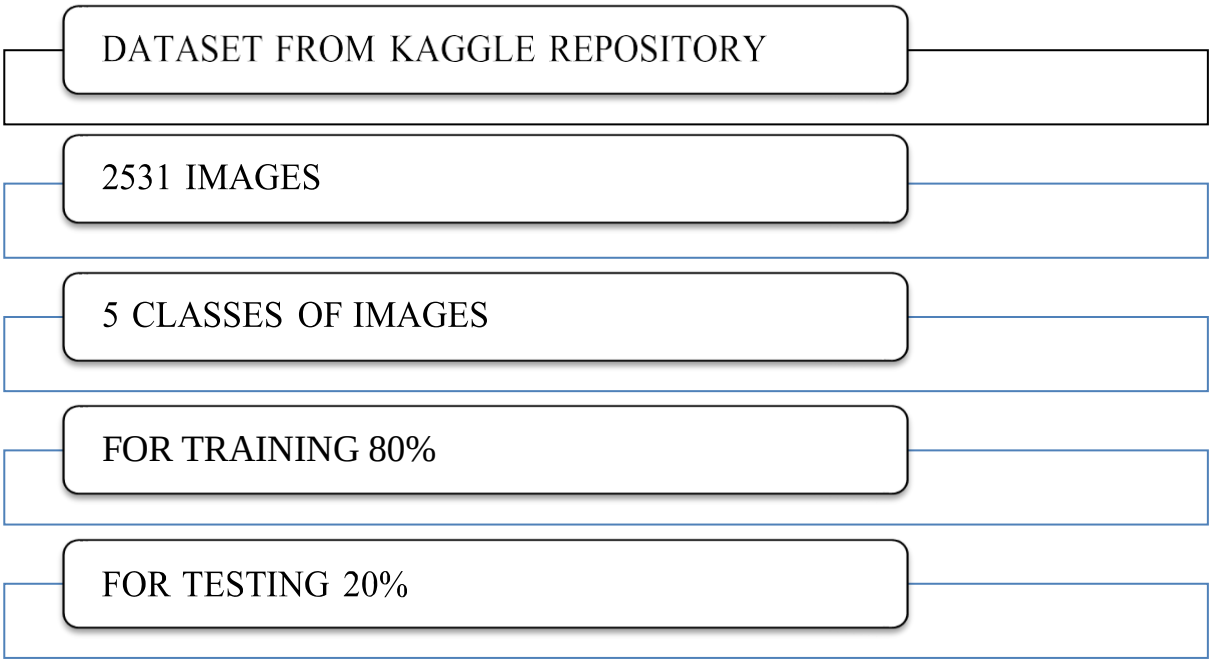


Fig.7.1.1 Data Collection

7.1.2 FIVE CLASSES OF IMAGES

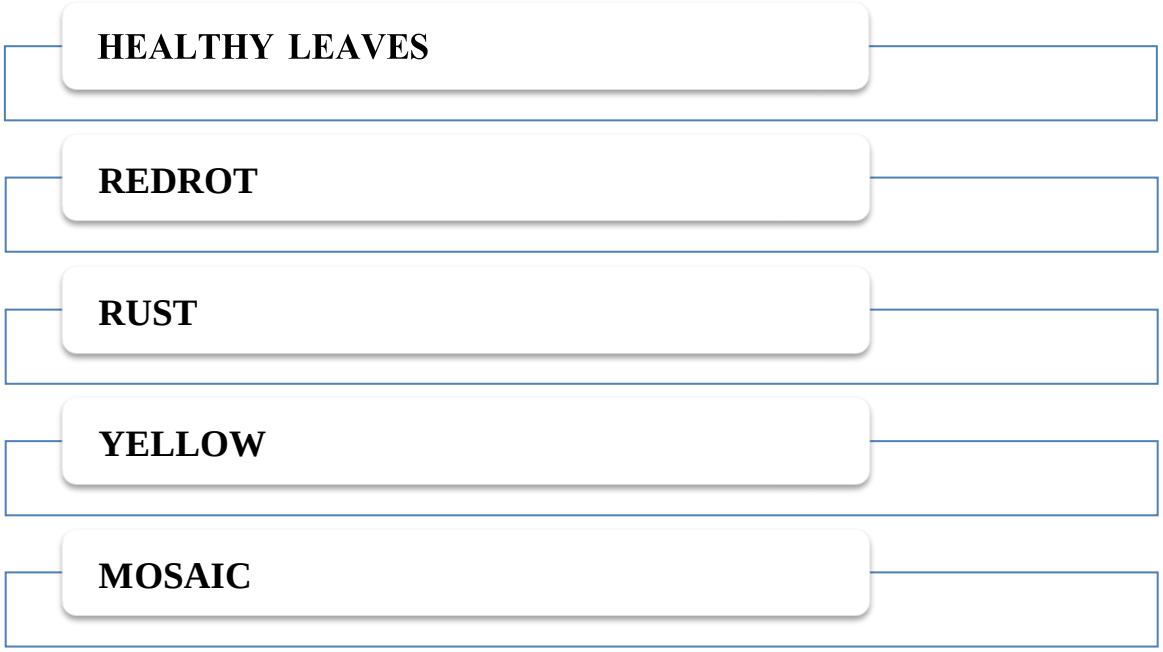


Fig.7.1.2 Five class of image

7.2 Rexnet - 150

We proposed our new disease detection system using Rank Expansion Networks (ReXNets) model, which follows the design principles. It turns out that a simple modification upon the baseline models could show remarkable improvement in performance on RexNet classification. The performance improvement of the RexNet classification is well transferred to the disease detection on Sugarcane leaf disease dataset from Kaggle repository and to the various fine-grained classification tasks, showing the effectiveness of our model as a strong feature extractor. Our contributions is the investigation of representational bottleneck problem that happens in a network through a mathematical and an empirical studies.

We apply our principles to Densenet. We found the accuracy improvements on Densenet (77.1%) while with similar computational costs. Verifying representational bottleneck in pretrained models. We now make a final backup by measuring the matrix rank of the output of each layer to reveal the representational bottleneck. Specifically, we use two Rexnet trained model that follows the design principles to visualize the cumulative distribution of the singular values computed with each feature set. Using randomly sampled 2,000 images in Rexnet validation set, we compute the singular values from the extracted features of 1) each layer after the nonlinearity in every inverted bottleneck and 2) after the nonlinearity atthe penultimate layer. We first normalize all singular values to [0, 1] to manage different singular values from different layers and then plot the cumulative percentage of normalized singular values for each layer.Heavy model comparison on other networks. The result of Rexnet model is compared with popular models. Note that ReXNets are trained and evaluated with the large dataset of images.

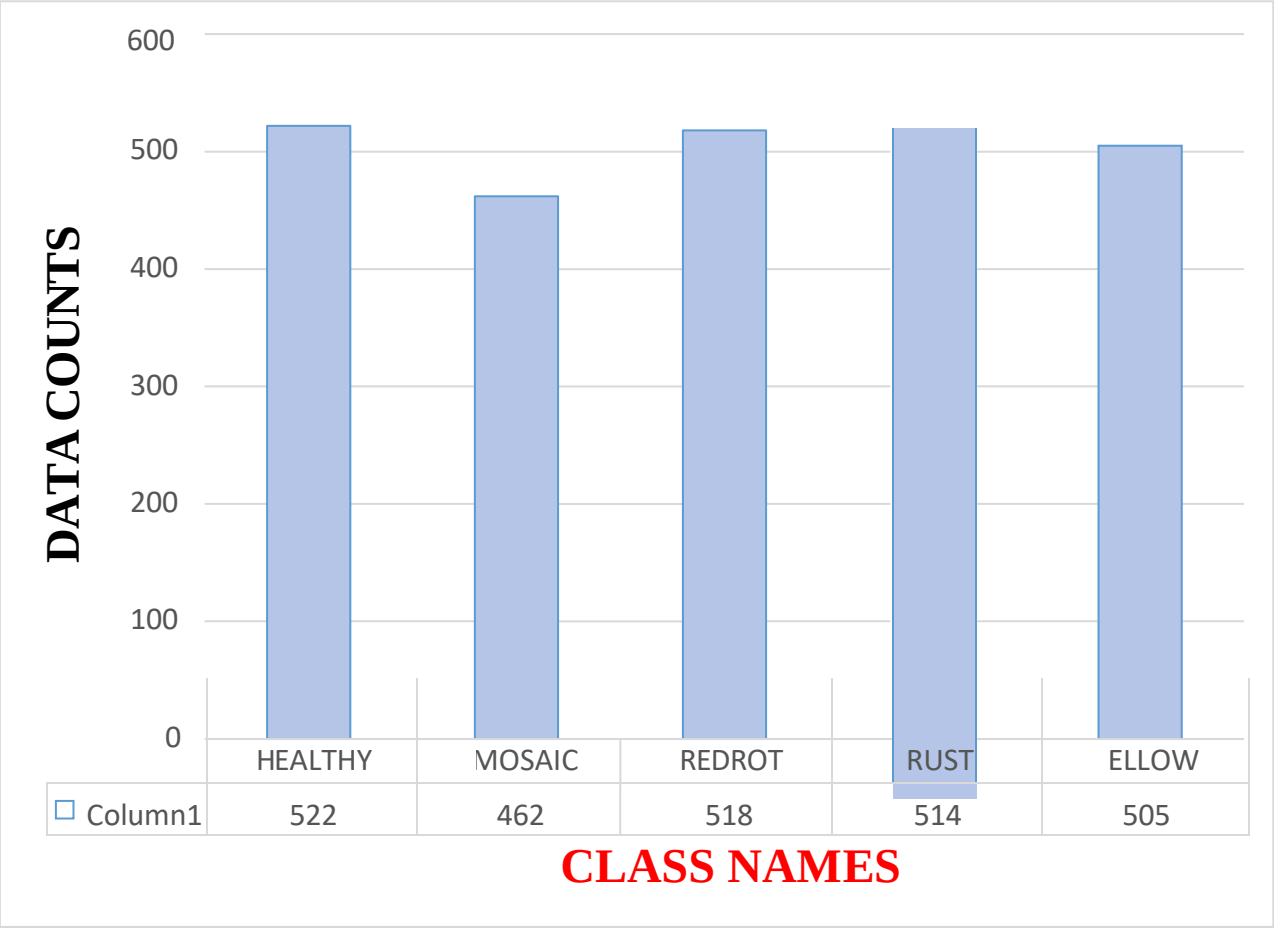


Fig 7.2 Dataset Imbalance Analysis

7.2.1 PERFORMANCE ANALYSIS

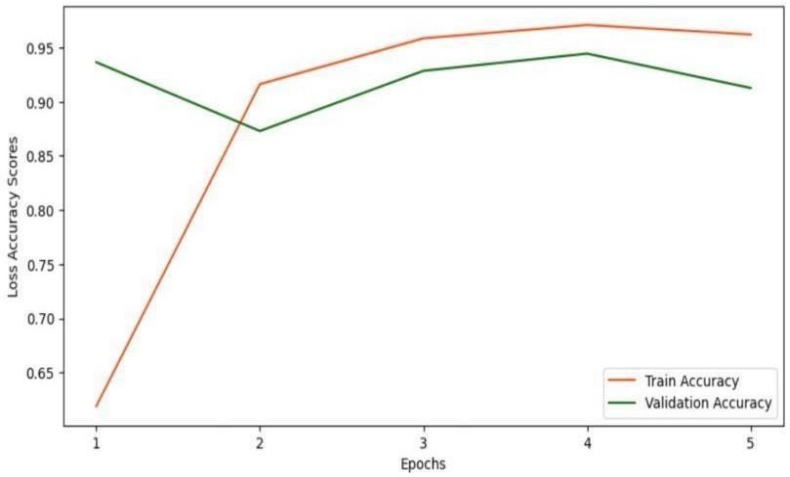


Fig 7.2.1 Accuracy Values

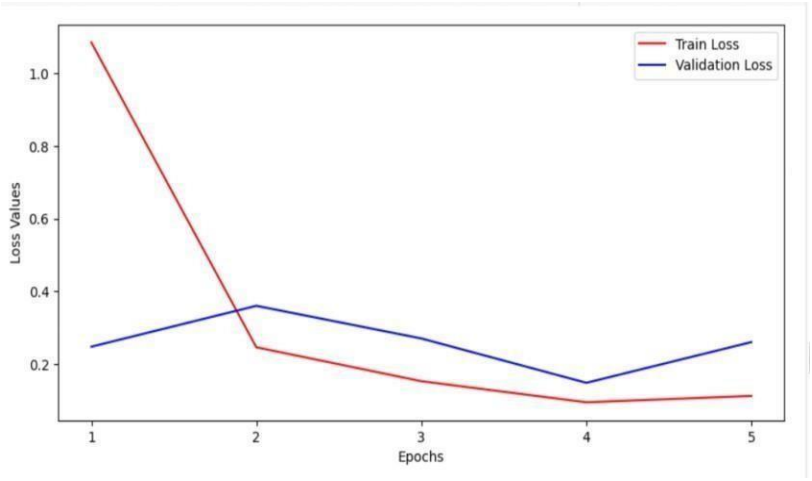


Fig 7.2.1 wsLoss Values

7.2.2 COMPARISON OF MODELS

Network Model	Top-1(%)	Top-5(%)	Datasets Averages
DenseNet	55.7	57.9	Dataset of 2531 Images
RexNet 150	91.3	96.9	Dataset of 2531 Images

Fig.7.2.2 Model comparison

CHAPTER - 8
CONCLUSION

8.1 CONCLUSION

Sugarcane leaf disease detection is a critical aspect of crop management practices, as it directly impacts the yield and quality of sugarcane crops. In recent years, the application of convolutional neural networks (CNNs) in disease detection has shown promising results, offering accurate and efficient methods for identifying and classifying sugarcane leaf diseases. Through the analysis of leaf images, CNNs can detect diseases at early stages, enabling timely interventions and improved crop management strategies.The utilization of CNNs for sugarcane leaf disease detection has led to significant advancements in agricultural technology. These advanced algorithms can accurately distinguish between healthy and diseased leaves, even when diseases are in their initial stages and not easily detectable by the human eye. By leveraging deep learning techniques, CNN models can learn complex patterns and features from large datasets of labeled leaf images, enabling them to generalize well to unseen data and effectively classify a wide range of diseases.

Early detection of sugarcane leaf diseases is crucial for preventing yield losses and minimizing the spread of diseases throughout sugarcane fields. CNN-based disease detection systems offer the potential to automate the process of disease identification, reducing the need for manual inspection and enabling real-time monitoring of crop health. This automation enhances efficiency and scalability, allowing farmers to monitor large areas of sugarcane fields with greater ease and accuracy.Despite the significant advancements in CNN-based disease detection, several challenges remain to be addressed. These challenges include the need for large labeled datasets, model generalization across different environmental conditions and regions, and the interpretability of model predictions.

Additionally, the deployment of CNN models in resource-constrained environments, such as rural agricultural areas, may require adaptations to ensure accessibility and usability by farmers. This includes developing user-friendly tools for farmers, integrating disease detection systems with precision agriculture techniques, and enhancing the interpretability and robustness of CNN models. By using RexNet-150 model the proposed system attain 97 percent accuracy. By continuing to the field of sugarcane leaf disease detection, we can contribute to the sustainable cultivation of sugarcane crops and ensure food security for future generations.

8.2 FUTURE ENHANCEMENTS

Future work in our proposed system to expand the website it includes educational resources, tutorials, and guides on sugarcane diseases, detection methods, and management practices. Providing users with access to relevant information empowers them to make informed decisions and take proactive measures to protect their crops. It also provide treatment for the respective disease by suggesting fertilizers.

.

CHAPTER – 9

APPENDIX

9.1 SOURCE CODE

#index.html#

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>Care for cane</title>
    <meta content="width=device-width, initial-scale=1.0" name="viewport" />
    <meta content="" name="keywords" />
    <meta content="" name="description" />
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Favicon -->
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href=".../static/img/apple-touch-icon.png."/>
    <link
      rel="icon"
      type="image/png"
      sizes="32x32"
      href="../static/img/favicon-32x32.png" />
    <link
      rel="icon"
      type="image/png
      sizes="16x16"
      href="../static/img/favicon-16x16.png"/>
```

```
    <link rel="manifest" href="../static/img/site.webmanifest" />
<link
      rel="mask-icon"
      href="../static/img/safari-pinned-tab.svg"
      color="#5bbad5"/>
    <meta name="msapplication-TileColor" content="#da532c" />
    <meta name="theme-color" content="#ffffff"
    <!-- Google Web Fonts -->
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link

href="https://fonts.googleapis.com/css2?family=Libre+Baskerville:wght@700
&family=Open+Sans:wght@400;500;600&display=swap"
      rel="stylesheet"/>
    <!-- Icon Font Stylesheet -->
    <link
      href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.10.0/css/all.min.css"
      rel="stylesheet"/>
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.4.1/font/bootstrap-
icons.css"
      rel="stylesheet"/>
    <!-- Libraries Stylesheet -->
    <link href="../static/lib/animate/animate.css" rel="stylesheet" />
    <link
      href="../static/lib/owlcarousel/owl.carousel.min.js"
      rel="stylesheet"/>
```

```
    <link href="../static/lib/lightbox/js/lightbox.min.js" rel="stylesheet" />
    <link rel="stylesheet" href="https://cdn.plyr.io/3.6.8/plyr.css" />
    <!-- Customized Bootstrap Stylesheet -->
    <link href="../static/css/bootstrap.min.css" rel="stylesheet" />
    <!-- Template Stylesheet -->
    <link href="../static/css/style.css" rel="stylesheet" />
    <link href="../static/css/style1.css" rel="stylesheet" />
  </head>
  <body>
    <!-- Spinner Start -->
    <div
      id="spinner"
      class="show bg-white position-fixed translate-middle w-100 vh-100 top-50
start-50 d-flex align-items-center justify-content-center">
      <div
        class="spinner-border text-primary"
        role="status"
        style="width: 3rem; height: 3rem"></div>
    </div>
    <nav
      class="navbar navbar-expand-lg navbar-dark sticky-top px-4 px-lg-5"
      style="background: rgb(16 52 19)">
      <a
        class="navbar-brand d-flex align-items-center navbar-logo"
        style="”>
        <img
          src="../static/img/logo.png"
          class="rounded-circle "
          width="50"
```

```
style="padding: 0%; margin: 0%"alt="
        <h3class="m-0 navbar-crop text-light style="">Care for cane</h3>
       <button
        type="button"
        class="navbar-toggler me-0"
        data-bs-toggle="collapse"
        data-bs-target="#navbarCollapse">
        <span class="navbar-toggler-icon"></span>
       </button>
       <div class="collapse navbar-collapse" id="navbarCollapse">
        <div class="navbar-nav ms-auto p-4 p-lg-0">
         <a href="../templates/base.html" class="nav-item nav-link text-
   light">Home</a>
         <a href="../templates/prediction.html" class="nav-item nav-link text-
   light">Prediction</a>
        </div
    <!-- <div class="border-start ps-4 d-none d-lg-block">
          <button type="button" class="btn btn-sm p-0"><i class="fa fa-
   search"></i></button>
         </div> -->
      </div>
    </nav>
    <!-- Navbar End -->
    <!-- Details about website -->
    <div class="container-xxl py-5" >
      <div class="container">
        <div class="row g-5 align-items-end">
          <div class="col-lg-6">
    <div class="row g-2">
```

```
                <div class="col-6 position-relative wow fadeIn" data-wow-
delay="0.7s">    <div class="about-experience rounded text-light"
style="background: rgb(16 52 19)" >
                    <div  align="center">
                      <h1  class="display-1 text-light  mb-0">Care  for  Cane</h1>
                      </div>
                   </div>
                </div>
                <div  class="col-6  wow  fadeIn"  data-wow-delay="0.1s">
                   <img  class="img-fluid  rounded"  src="../static/img/d1.png">
                </div>
                <div  class="col-6  wow  fadeIn"  data-wow-delay="0.3s">
                   <img  class="img-fluid  rounded"  src="../static/img/Red
Rot.jpg">
                </div>
                <div  class="col-6  wow  fadeIn"  data-wow-delay="0.5s">
                    <img  class="img-fluid  rounded"
src="../static/img/leaf_rust.jpg">
                </div>
              </div>
            </div>
            <div  class="col-lg-6  wow  fadeIn"  data-wow-delay="0.5s">
              <!-- <p class="section-title bg-white text-start text-primary pe-
3">About Us</p> -->
              <h1 class="mb-4">Know  About  Sugarcane  Diseases</h1>
<p class="mb-4"><a href="https://agri.sindh.gov.pk/diseases-of- sugarcane-
their-control" target="_blank">More than 50 diseases are reported in
sugarcane</a>, fungi, bacteria, viruses and nematodes cause the most
destructive diseases. Theseall diseases are injurious in some areas, in some
years and on some plant parts. All parts ofplant are subject to disease and one or
more diseases can occur on virtually every plant and inevery field.</p>
```

```
<div class="row g-5 pt-2 mb-5">
                    <div class="col-sm-6">
                        <img class="img-fluid mb-4" src="../static/img/service.png" alt="">
                        <h5 class="mb-3">Dedicated Services</h5>
                        <span>Sugarcane Diseases Detector</span>
                    </div>
                </div>
                <a class="btn btn-outline-success rounded-pill py-3 px-5" href="../templates/service.html">Explore More</a>
            </div>
        </div>
    </div>
</div>
<div class="container-xxl py-5">
  <div class="container">
      <div class="text-center mx-auto pb-4 wow fadeInUp" data-wow-delay="0.1s" style="max-width: 500px;">
          <h2 class="section-title bg-white text-center text-primary px-3">Diseases</h2>
          <br><br>
      </div>
      <div class="row gy-5 gx-4">
          <div class="col-lg-4 col-md-6 pt-5 wow fadeInUp" data-wow-delay="0.1s">
              <div class="service-item d-flex h-100">
                  <div class="service-img">
```

```
                <img class="img-fluid" src="../static/img/Red Rot.jpg" alt="">
            </div>
            <div class="service-text p-5 pt-0">
                <div class="service-icon">
                    <img class="img-fluid rounded-circle" src="../static/img/Red Rot.jpg" alt="">
                </div>
                <h5 class="mb-3">Red rot</h4>
                <p class="mb-4">The disease causes wilting of canes. The affected canes show drying of leaves from top to bottom.
                </p>
            </div>
        </div>
    </div>
    <div class="col-lg-4 col-md-6 pt-5 wow fadeInUp" data-wow-delay="0.1s">
        <div class="service-item d-flex h-100">
            <div class="service-img">
                <img class="img-fluid" src="../static/img/Red_Rust.jpg" alt="">
            </div>
            <div class="service-text p-5 pt-0">
                <div class="service-icon">
                    <img class="img-fluid rounded-circle" src="../static/img/Red_Rust.jpg" alt="">
                </div>
                <h5 class="mb-3">Red rust</h4>
                <p class="mb-4">The disease causes wilting of canes. The affected canes show drying of leaves from top to bottom.
                </p>
            </div>
```

```
            </div>
          </div>
          <div class="col-lg-4 col-md-6 pt-5 wow fadeInUp" data-wow-
delay="0.1s">
            <div class="service-item d-flex h-100">
              <div class="service-img">
                <img class="img-fluid" src="../static/img/Yellow Leaf
Virus.jpg" alt="">
              </div>
              <div class="service-text p-5 pt-0">
                <div class="service-icon">
                  <img class="img-fluid rounded-circle"
src="../static/img/Yellow Leaf Virus.jpg" alt="">
                </div>
                <h5 class="mb-3">Yellow leaf virus</h>
                 <p class="mb-4">The disease causes wilting of canes. The
affected canes show drying of leaves from top to bottom.
                </p>
              </div>
            </div>
          </div>
        <div align="center">
          <div class="col-lg-4 col-md-6 pt-5 wow fadeInUp" data-wow-
delay="0.3s">
            <div class="service-item d-flex h-100">
              <div class="service-img">
                <img class="img-fluid" src="../static/img/Bacterial Blight.jpg"
alt="">
              </div>
              <div class="service-text p-5 pt-0">
                <div class="service-icon">
```

```
                    <img class="img-fluid rounded-circle"
src="../static/img/Bacterial Blight.jpg" alt="">
                </div>
                <h5  class="mb-3">Mosaic</h5>
                <p class="mb-4">Mottling of young crown leaves showing a
definite pattern of alternating dark and light green coloured patches of varying size
and run parallel to the midrib of leaf. </p>
            </div>
        </div>
    </div>
</div>
</div>
 <!-- JavaScript Libraries -->
 <script  src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
 <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0/dist/js/bootstrap.bundle.min.j
s"></script>
 <script   src="https://cdn.plyr.io/3.6.8/plyr.js"></script>
 <script    src="https://cdn.plyr.io/3.6.8/plyr.youtube.js"></script>
 <script>
   const player = new Plyr("#player", {
     youtube: { noCookie: true, rel: 0, showinfo: 0, iv_load_policy: 3 },
   });
 </script>
 <script src=".
./static/lib/wow/wow.js"></script>
 <script   src="../static/lib/easing/easing.js"></script>
 <script  src="../static/lib/waypoints/waypoints.min.js"></script>
```

```
<script    src="../static/lib/owlcarousel/owl.carousel.js"></script>
<script    src="../static/lib/counterup/counterup.min.js"></script>
<script     src="../static/lib/parallax/parallax.min.js"></script>
<script     src="../static/lib/lightbox/js/lightbox.js"></script>
<script    src="../static/js/image_upload.js"></script>
<script   src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4"></script>
<script>
  (function ($) {
  "use strict";
    // Spinner
    var spinner = function () {
      setTimeout(function () {
        if ($("#spinner").length > 0) {
          console.log("its running");
          $("#spinner").removeClass("show");
        }
      }, 1);
    };
    spinner();
    // Initiate the wowjs
    new WOW().init();
    // Sticky Navbar
    $(window).scroll(function () {
      if ($(this).scrollTop() > 300) {
        $(".sticky-top").addClass("shadow-sm").css("top", "0px");
      } else {
        $(".sticky-top").removeClass("shadow-sm").css("top", "-100px");
      }
    }
```

```
// Back to top button
$(window).scroll(function () {
  if ($(this).scrollTop() > 100) {
    $(".back-to-top").fadeIn("slow");
  } else {
    $(".back-to-top").fadeOut("slow");
  }
});
$(".back-to-top").click(function () {
  $("html, body").animate({ scrollTop: 0 }, 1500, "easeInOutExpo");
  return false;
});
// Facts counter
$('[data-toggle="counter-up"]').counterUp({
  delay: 10,
  time: 2000,
});
// Testimonials carousel
$(".testimonial-carousel").owlCarousel({
  autoplay: true,
  smartSpeed: 1000,
  items: 1,
  dots: false,
  loop: true,
  nav: true,
  navText: [
    '<i class="bi bi-chevron-left"></i>',
    '<i class="bi bi-chevron-right"></i>',
  ],
```

```
    });
  })(jQuery);
  function progress() {
    $(".progress_container").css("visibility", "visible");
  }
</script>
<script>
  var prediction_data = {{Disease_classification}};
  var total = prediction_data.reduce(function(a, b) { return a + b; }, 0);
  options:      });
}
</script>
<script>
  // Get the histogram data from the context variable
  const histData = {{ hist|safe }};
  const bins = {{ bins|safe }};
  const equHistData = {{ equ_hist|safe }};
  const equBins = {{ equ_bins|safe }};
  // Create the histograms using Chart.js
  const histChart = new
Chart(document.getElementById('histogram').getContext('2d'), {
    type: 'bar',
    data: {
      labels: bins,
      datasets: [{
        label: 'Grayscale Image',
        data: histData,
        backgroundColor: '#5B8C51',
        borderColor: '5B8C51',
```

```
      borderWidth: 1
    }]
  },
  options: {
    scales: {
          yAxes: [{
            ticks: {
            max: 1500
            }
          }],

          xAxes: [{
            ticks: {
              autoSkip: true,
              maxTicksLimit: 20
            }
          }]
    }
  }
})
xAxes: [{
        ticks: {
          autoSkip: true,
          maxTicksLimit: 20
        }
      }]
    }
  }
});
```

```
</script>
<script>
  var ctx  =  document.getElementById('myChart_contour').getContext('2d');
  var myChart = new Chart(ctx, {
    type: 'line',
    data: {
      labels: Array.from({length: 200}, (_, i) => i), // create an array with 0 to 200 values
      datasets: [{
        label: 'Number of Contours',
        data: Array.from({length: 200}, (_, i) => i === {{Num_of_contour}} ? i : null), // create an array with 48 as the value and null for all other values
        borderColor: '#5b8c51',
        borderWidth: 1,
        fill: false
      }]
    },
    options:  {
      scales:  {
      yAxes: [{
          ticks:                {
            beginAtZero: true
          }
        }]
      },
      responsive: true,
      maintainAspectRatio: false
    }
  });
```

```
  </script>
 </body>
</html>
```

#prediction.html#

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta  charset="utf-8" />
    <title>Care  for  cane</title>
    <meta  content="width=device-width,  initial-scale=1.0" name="viewport" />
    <meta content="" name="keywords" />
    <meta  content=""  name="description"  />
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- Favicon -->
    <link
      rel="apple-touch-icon"
      sizes="180x180"
      href=".../static/img/apple-touch-icon.png."
    />
    <link
      rel="icon"
      type="image/png"
      sizes="32x32"
      href="../static/img/favicon-32x32.png"
    />
    <link
      rel="icon"
      type="image/png"
      sizes="16x16"
```

```
    href="../static/img/favicon-16x16.png"
  />
  <link rel="manifest" href="../static/img/site.webmanifest" />
  <link
    rel="mask-icon"
    href="../static/img/safari-pinned-tab.svg"
    color="#5bbad5"
  />
  <meta name="msapplication-TileColor" content="#da532c" />
  <meta name="theme-color" content="#ffffff" />

  <!-- Google Web Fonts -->
  <link rel="preconnect" href="https://fonts.googleapis.com" />
  <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
  <link

href="https://fonts.googleapis.com/css2?family=Libre+Baskerville:wght@700
&family=Open+Sans:wght@400;500;600&display=swap"
    rel="stylesheet"
  />
  <!-- Icon Font Stylesheet -->
  <link
    href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/5.10.0/css/all.min.css
    rel="stylesheet"
  />
  <link
```

```
    href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.4.1/font/bootstrap-icons.css"
    rel="stylesheet"
  />
  <!-- Libraries Stylesheet -->
  <link href="../static/lib/animate/animate.css" rel="stylesheet" />
  <link
    href="../static/lib/owlcarousel/owl.carousel.min.js"
    rel="stylesheet"
  />
  <link href="../static/lib/lightbox/js/lightbox.min.js" rel="stylesheet" />
  <link rel="stylesheet" href="https://cdn.plyr.io/3.6.8/plyr.css" />
  <!-- Customized Bootstrap Stylesheet -->
  <link href="../static/css/bootstrap.min.css" rel="stylesheet" />

  <!-- Template Stylesheet -->
  <link href="../static/css/style.css" rel="stylesheet" />
  <link href="../static/css/style1.css" rel="stylesheet" />
 </head>
 <body>
  <!-- Spinner Start -->
  <div
    id="spinner"
    class="show bg-white position-fixed translate-middle w-100 vh-100 top-50 start-50 d-flex align-items-center justify-content-center"
  >
<div
class="spinner-border text-primary"
      role="status"
```

```
      style="width: 3rem; height: 3rem"
    ></div>
  </div>
  <nav
  class="navbar navbar-expand-lg navbar-dark sticky-top px-4 px-lg-5"
  style="background: rgb(16 52 19)"
>
  <a
    class="navbar-brand d-flex align-items-center navbar-logo"
    style""
    <img
      src="../static/img/logo.png"
      class="rounded-circle "
      width="50"
      style="padding: 0%; margin: 0%"
      alt="
    <h3class="m-0 navbar-crop text-light style"">Care for cane</h3>
    <button
      type="button"
      class="navbar-toggler me-0"
      data-bs-toggle="collapse"
      data-bs-target="#navbarCollapse"
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarCollapse">
      <div class="navbar-nav ms-auto p-4 p-lg-0">
        <a href="../templates/index.html" class="nav-item nav-link text-
light">Home</a>
```

```
        <a href="../templates/prediction.html" class="nav-item nav-link text-light">Prediction</a>
       <!-- <div class="border-start ps-4 d-none d-lg-block">
            <button type="button" class="btn btn-sm p-0"><i class="fa fa-search"></i></button>
          </div> -->
     </div>
   </nav>
   <!-- Navbar End -->
   <!-- Footer Start -->
<link href="../static/css/image_upload.css" rel="stylesheet" />
<div
  class="progress_container"
  style="
    visibility: hidden;
    position: fixed;
    width: 100%;
    height: 100%;
    background: rgba(36, 36, 36, 0.8);
    top: 0;
    left: 0;
    z-index: 9999;
    filter: blur(45%);
  "
>
  <div
    class="d-flex justify-content-center"
    style="top: 50%; margin: 0px 35%; width: 30%; position: relative"
  >
```

```
    <div
      id=""
      class="spinner-border text-success"
      style="width: 5rem; height: 5rem"
      role="status"
    ></div>
  </div>
</div>
<div
  class="container-fluid banner px-0"
  data-parallax="scroll"
  style="background: rgb(255, 255, 255)
    <div class="container py-5 mt-3">
      <div class="row g-5 py-5">
        <div
          class="col-lg-12 wow fadeIn"
          data-wow-delay="0.2s"
          style="margin: auto; background: rgb(255, 255, 255); padding: 30px
30px"
          <div class="row g-4 align-items-center">
            <div class="col-lg" >
              <h2 class="mb-5">Sugarcane Diseases Detector</h2>
              <p style="text-align: justify">
                Fast and Accurate Diagnosis for Healthier Crops. Our AI-powered
                technology can quickly detect and identify diseases in sugarcane
                crops, helping farmers take proactive measures to protect their
                harvests and increase yields. Try our easy-to-use platform today
                and experience the benefits of cutting-edge agricultural
                innovation.
```

```
  </p>
</div>
<div class="col-lg">
  <form
    enctype="multipart/form-data"
    method="GET"
    action="../disease_detection.py"
    <div class="row g-3">
      <center>
        <div  class="image_upload">
          <div  class="wrapper">
            <header></header>
            <span>
              <input
                class="file-input"
                type="file"
                accept="image/*"
                name="original_image"
                id="original_image"
                required
                onchange="showPreview(event);"
                id="file-ip-1 subject original_image"
                style="display: none
               <label
                style="color: #5b8c51; cursor: pointer;"
                class="fa fa-cloud-upload-alt fa-3x"
                for="original_image"
              ></label>
            </span>
```

```
<p>Browse File to Upload
  <p style="color:red"></p>
        <section class="progress-area"></section>
<section class="uploaded-area"></section>
<div class="preview col-6 pb-3">
  <img
    id="file-ip-1-preview"
    class="img-fluid rounded"
    width="330"
  />
</div>
</div>
</div>
<div id="process" style="display: none">
  <button>
    class="btn btn-outline-primary py-3 px-5 mt-3"
    onclick="progress()"
    type="submit" action="sugarcane.py
    cursor: pointer;"
                Detect Disease
  </button>
</iframe>
</div>
</center>
</div>
</form>
</div>
</div>
</div>
```

```
  </div>
 </div>
 <div class="container >
      <a
        class="btn btn-primary py-3 px-5 btn-small col-sm-3 m-3"
        onclick="window.print()"
        style="cursor: pointer"
        >Save Report</a
       >
      </div>
  </div>
 </div>
</div>
<div
    class="container-fluid footer mt-5 py-5 wow fadeIn"
    style="background: rgb(16 52 19)"
    data-wow-delay="0.1s”>
    <div class="container py-5">
     <div class="row g-5">
      <div class="col-lg-4 col-md-6">
       <h5 class="text-white mb-4">Our college</h5>
       <p class="mb-2">
        <i class="fa fa-map-marker-alt me-3"></i>panruti
       </p>
       <p class="mb-2">
        <i class="fa fa-phone-alt me-3"></i>4226
       </p>
       <p class="mb-2">
        <i class="fa fa-envelope me-3"></i>ucep.gmail.com
```

```
          </p>
          <div class="d-flex pt-3>
          </div>
        </div>
        <div class="col-lg-4 col-md-6">
          <h5 class="text-white mb-4">Quick Links</h5>
          <a class="btn btn-link"
href="../templates/prediction.html">Prediction</a>
        </div>
        <div class="col-lg-4 col-md-6">
          <h5 class="text-white mb-4">Newsletter</h5>
          <p>
            An automated disease detection system is needed to help
            distinguish plant infections by the plant's appearance and visual
            manifestations could be of incredible assistance to novices in the
            horticultural cycle..
       </p>
        </div>
       </div>
     </div>
   </div>
   <!-- Footer End -->
   <!-- Copyright Start -->
   <div class="container-fluid text-body copyright py-2">
    <div class="container">
      <div class="row">
        <div class="col-md-6 text-center text-md-start mb-3 mb-md-0">
          &copy; <a class="fw-semi-bold" href="#">ucep</a>, All Right
          Reserved.
```

```
        </div>
      </div>
    </div>
  </div>
  <!-- Copyright End -->
  <!-- Back to Top -->
  <a
    href="#"
    class="btn btn-lg btn-primary btn-lg-square rounded-circle back-to-top"
    ><i class="bi bi-arrow-up"></i
  ></a>
  <!-- JavaScript Libraries -->
  <script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0/dist/js/bootstrap.bundle.min.j
s"></script>
  <script src="https://cdn.plyr.io/3.6.8/plyr.js"></script>
  <script src="https://cdn.plyr.io/3.6.8/plyr.youtube.js"></script>
  <script>
    const player = new Plyr("#player", {
      youtube: { noCookie: true, rel: 0, showinfo: 0, iv_load_policy: 3 },
    });
  </script>
  <script src="../static/lib/wow/wow.js"></script>
  <script src="../static/lib/easing/easing.js"></script>
  <script src="../static/lib/waypoints/waypoints.min.js"></script>
  <script src="../static/lib/owlcarousel/owl.carousel.js"></script>
  <script src="../static/lib/counterup/counterup.min.js"></script>
  <script src="../static/lib/parallax/parallax.min.js"></script>
```

```
<script     src="../static/lib/lightbox/js/lightbox.js"></script>
<script    src="../static/js/image_upload.js"></script>
<script   src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4"></script>
<script>
  (function ($) {
  "use strict";
    // Spinner
    var spinner = function () {
      setTimeout(function () {
        if ($("#spinner").length > 0) {
          console.log("its running");
          $("#spinner").removeClass("show");
        }
      }, 1);
    };
    spinner();
    // Initiate the wowjs
    new WOW().init();
    // Sticky Navbar
    $(window).scroll(function () {
      if ($(this).scrollTop() > 300) {
        $(".sticky-top").addClass("shadow-sm").css("top", "0px");
      } else {
        $(".sticky-top").removeClass("shadow-sm").css("top", "-100px");
      }
    });
    // Back to top button
    $(window).scroll(function () {
      if ($(this).scrollTop() > 100) {
```

```
    $(".back-to-top").fadeIn("slow");
   } else {
    $(".back-to-top").fadeOut("slow");
   }
  });
  $(".back-to-top").click(function () {
   $("html, body").animate({ scrollTop: 0 }, 1500, "easeInOutExpo");
   return false;
  });
  // Facts counter
  $('[data-toggle="counter-up"]').counterUp({
   delay: 10,
   time: 2000,
  });
  // Testimonials carousel
  $(".testimonial-carousel").owlCarousel({
   autoplay: true,
   smartSpeed: 1000,
   items: 1,
   dots: false,
   loop: true,
   nav: true,
   navText: [
    '<i class="bi bi-chevron-left"></i>',
    '<i class="bi bi-chevron-right"></i>',
   ],
  });
})(jQuery);
```

```
    function progress() {
      $(".progress_container").css("visibility", "visible");
    }
  </script>
  <script>
    var prediction_data = {Disease_classification};
    var total = prediction_data.reduce(function(a, b) { return a + b; }, 0);
        // Create the chart
    var chart = new Chart(ctx, {
        type: 'pie',
        data: {
            datasets: [{
                data: [normalized_data[4]*100, normalized_data[7]*100,
normalized_data[1]*100, normalized_data[2]*100, normalized_data[6]*100,
normalized_data[5]*100, normalized_data[0]*100, normalized_data[3]*100],
                backgroundColor: ["#FF6384",
"#36A2EB","#FFCE56","#4BC0C0","#9966FF","#FF9F40","#FDB45C","#FF6
B6B"]
            }]
        },
        options: {
            title: {
                display: true,
                text: 'Distribution of Sugarcane Leaf Diseases Detected'
            },
            legend: {
                display: true,
                position: 'bottom',
            },
            animation: {
```

```
            animateRotate: true,
            animateScale: false,
        },
        tooltips: {
            callbacks: {
                label: function(tooltipItem, data) {
                    var dataset = data.datasets[tooltipItem.datasetIndex];
                    var label = data.labels[tooltipItem.index];
                    var value = dataset.data[tooltipItem.index];
                    return label + ': ' + value.toFixed(2) + '%';
                }
            }
        }
    }
  });
</script>
<script>
  // Get the histogram data from the context variable
  const histData = {histsafe };
  const bins = {binssafe };
  const equHistData = {equ_histsafe};
  const equBins = {equ_binssafe};
  // Create the histograms using Chart.js
  const histChart = new
Chart(document.getElementById('histogram').getContext('2d'), {
    type: 'bar',
    data: {
    labels: bins,
    datasets: [{
```

```
        label: 'Grayscale Image',
        data: histData,
        backgroundColor: '#5B8C51',
        borderColor: '5B8C51',
        borderWidth: 1
      }]
    },
    options: {
      scales: {
          yAxes: [{
            ticks: {
            max: 1500
            }
          }],
          xAxes: [{
            ticks: {
              autoSkip: true,
              maxTicksLimit: 20
            }
          }]
      }
    }
  });
  });
</script>
<script>
  var ctx = document.getElementById('myChart_contour').getContext('2d');
  var myChart = new Chart(ctx, {
    type: 'line',
```

```
        data: {
          labels: Array.from({length: 200}, (_, i) => i), // create an array with 0 to 200 values
          datasets: [{
            label: 'Number of Contours',
            data: Array.from({length: 200}, (_, i) => i ={Num_of_contour} ? i : null), // create an array with 48 as the value and null for all other values
            borderColor: '#5b8c51',
            borderWidth: 1,
            fill: false
          }]
        },
        options:  {
          scales:  {
          yAxes: [{
              ticks:                {
                beginAtZero: true
              }
            }]
          },
          responsive: true,
          maintainAspectRatio: false
      });
    </script>
  </body>
</html>
```

#style1.css#

```
.navbar .dropdown-toggle::after {
    border: none;
    content:  "\f107";
```

```
  font-family: "Font Awesome 5 Free";
  font-weight: 900;
  vertical-align: middle;
  margin-left: 8px;
}

@media (max-width: 991.98px) {
  .navbar .navbar-nav .nav-link {
    margin-right: 0;
    padding: 10px 0;
  }
  .navbar .navbar-nav {
    border-top: 1px solid #eeeeee;
  }
}
@media (min-width: 992px) {
  .navbar .nav-item .dropdown-menu {
    display: block;
    border: none;
    margin-top: 0;
    top: 150%;
    opacity: 0;
    visibility: hidden;
    transition: 0.5s;
  }
  .navbar .nav-item:hover .dropdown-menu {
    position: absolute;
    width: 100%;
    height: 100%;
    object-fit: cover;
  }
}
.page-header .breadcrumb-item + .breadcrumb-item::before {
  color: var(--light);
}
.page-header .breadcrumb-item,
.page-header .breadcrumb-item a {
  font-size: 18px;
  color: var(--light);
}
/*** Section Title ***/
.section-title {
  position: relative;
```

```
  display: inline-block;
  text-transform: uppercase;
  font-weight: 600;
}
.section-title::before {
  position: absolute;
  content: "";
  width: calc(100% + 80px);
  height: 2px;
  top: 18px;
  left: -40px;
  background: var(--primary);
  z-index: -1;
}
.section-title::after {
  position: absolute;
  content: "";
  width: calc(100% + 120px);
  height: 2px;
  bottom: 6px;
  left: -60px;
  background: var(--primary);
  z-index: -1;
}
.section-title.text-start::before {
  width: calc(100% + 40px);
  left: 0;
}
.service-item .service-img img {
}
.service-item:hover .service-img::after {
  width: 0;
  left: auto;
  right: 0;
}
```

#image upload.js#

```
.service-item .service-text .service-icon {
  width: 140px;
  height: 140px;
  padding: 15px;
  margin-top: -70px;
  margin-bottom: 40px;
  border: 2px solid #5b8c51;
```

```
  background: #ffffff;
  border-radius: 50%;
  overflow: hidden;
  box-shadow: 0 0 60px rgba(0, 0, 0, 0.1);
  }
.service-item .service-text h5,
.service-item .service-text p {
  transition: 0.5s;
}
.service-item:hover .service-text h5,
.service-item:hover .service-text p {
  color: #ffffff;

.service-item .service-text .btn {
  color: #5b8c51;
  background: #ffffff;
  box-shadow: 0 0 45px rgba(0, 0, 0, 0.25);
}
.service-item .service-text .btn:hover {
  color: white;
  background: #5b8c51;
}

/*** Product ***/
.product-item {
  position: relative;
  border-radius: 8px;
  overflow: hidden;
  box-shadow: 0 0 45px rgba(0, 0, 0, 0.07);
}
.product-item .product-overlay {
  position: absolute;
  width: 100%;
  height: 100%;
  top: 0;
  left: 0;
  display: flex;
  align-items: center;
  justify-content: center;
  background: rgba(0, 0, 0, 0.5);
  opacity: 0;
  padding-top: 60px;
  transition: 0.5s;
```

```
}
.product-item:hover .product-overlay {
  opacity: 1;
  padding-top: 0;
}
/*** Team ***/
.team-item {
  position: relative;
  text-align: center;
  border-radius: 8px;
  box-shadow: 0 0 45px rgba(0, 0, 0, 0.07);
}
.team-item .btn {
  border-color: transparent;
  box-shadow: 0 0 45px rgba(0, 0, 0, 0.2);
}
/*** Testimonial ***/
.testimonial-img  {
  position:  relative;
  min-height: 400px;
}
.testimonial-img::after {
  position: absolute;
  content: "\f10d";
  font-family: "Font Awesome 5 Free";
  font-weight: 900;
  font-size: 200px;
  color: #eeeeee;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
  z-index: -1;
}
.testimonial-img img {
  position: absolute;
  width: 100px;
  height: 100px;
  border-radius: 100px;
}

.testimonial-img img:nth-child(1) {
  top: 0;
  left: 0;
```

```
}
.testimonial-img img:nth-child(2) {
  top: 60%;
  left:  20%;
}
.testimonial-img img:nth-child(3) {
  top: 20%;
  left:  60%;
}
  color:  #b0b9ae;
}
.footer .btn.btn-link {
  display: block;
  margin-bottom: 5px;
  padding: 0;
  text-align: left;
  color: #b0b9ae;
  font-weight:  normal;
  text-transform: capitalize;
  transition: 0.3s;
}
.footer .btn.btn-link::before {
  position: relative;
  content:  "\f054";
  font-family: "Font Awesome 5 Free";
  font-weight: 900;
  margin-right:  10px;
}
.footer  .btn.btn-link:hover  {
  color: var(--light);
  letter-spacing: 1px;
  box-shadow:  none;
}
.hidden {
  display: none;
}
.treatment_li {
  list-style:  none;
}
.treatment_li::before {
  content: "\00BB";
}
/* dropdown styling */
```

```
 .dropdown {
   position:  block;
 }
 .dropdown:hover .dropdown-menu {
   display: block;
 }
 .dropdown-menu {
   display: none;
   position: block;
   top: 100%;
   left: 0%;
   z-index: 1000;
   min-width: 8rem;
   padding: 0.5rem 0;
   font-size: 1rem;
   color: #212529;
   text-align: left;
   background-color: #fff;
   background-clip: padding-box;
   border: 1px solid rgba(0, 0, 0, 0.15);
   border-radius: 0.25rem;
 }
 .dropdown-item {
   display: block;
   width: 100%;
   padding: 0.25rem 1.5rem;
   clear: both;
   font-weight: 400;
   color: #212529;
   text-align: inherit;
   white-space: nowrap;
   background-color: transparent;
   border: 0;
 }
 .dropdown-item:focus,
 .dropdown-item:hover {
   color: #16181b;
   text-decoration: none;
   background-color: #f8f9fa;
 }
   url(path/to/nyt-cheltenham.woff) format("woff");
}
#piechart {
```

```
  width: 100%;
  height: auto;
}
::selection{
  color: #fff;
  background: #5B8C51;
}
.wrapper{
  width: 90%;
  max-width: 600px;
  margin: auto;
  background: #fff;
  border-radius: 10px;
  padding: 30px;
  box-shadow: 10px 10px 10px 10px rgba(0, 0, 0, 0.05);
}
.wrapper header{
  color: #5B8C51;
  font-size: 27px;
  font-weight: 600;
  text-align: center;
}
.wrapper form{
  height: 167px;
  display: flex;
  cursor: pointer;
  margin: 30px 0;
  align-items: center;
  justify-content: center;
  flex-direction: column;
  border-radius: 5px;
  border: 2px dashed #5B8C51;
}
form :where(i, p){
  color: #5B8C51;
}
form i{
  font-size: 50px;
}
form p{
  margin-top: 15px;
  font-size: 16px;
}
```

```
section .row{
  margin-bottom: 10px;
  background: #E9F0FF;
  list-style: none;
  padding: 15px 20px;
  border-radius: 5px;
  display: flex;
  align-items: center;
  justify-content: space-between;
}
section .row i{
  color: #5B8C51;
  font-size: 30px;
}
section .details span{
  font-size: 14px;
}
.progress-area .row .content{
  width: 100%;
  margin-left: 15px;
}
.progress-area .details{
  display: flex;
  align-items: center;
  margin-bottom: 7px;
  justify-content: space-between;
}
.progress-area .content .progress-bar{
  height: 6px;
  width: 100%;
  margin-bottom: 4px;
  background: #fff;
  border-radius: 30px;
}
.content .progress-bar .progress{
  height: 100%;
  width: 0%;
  background: #5B8C51;
  border-radius: inherit;
}
.uploaded-area{
  max-height: 232px;
  overflow-y: scroll;
```

```
}
.uploaded-area.onprogress{
  max-height: 150px;
}
.uploaded-area::-webkit-scrollbar{
  width: 0px;
}
.uploaded-area .row .content{
  display: flex;
  align-items: center;
}
.uploaded-area  .row  .details{
  display: flex;
  margin-left: 15px;
  flex-direction: column;
}
.uploaded-area  .row  .details  .size{
  color: #404040;
  font-size:  11px;
}
.uploaded-area i.fa-check{
  font-size: 16px;
}
@media (max-width: 1225px) {
  #histogram .chartjs-size-monitor,
  #histogram .chartjs-size-monitor-expand,
  #histogram  .chartjs-size-monitor-shrink,
  .chartjs-hidden-iframe,
  .y-axis-label,
  .x-axis-label {
    display: none !important;
  }
}
```

#**app.py**#

```
from flask import Flask, render_template, request
import os

app = Flask(_name_)

# Function to process uploaded images
def detect_disease(image) "Healthy","Redrot","Redrust","Yellow
leaf","Mosaic" Placeholder result
```

```
@app.route('/')
def home():
    return   render_template('index.html')

@app.route('/result', methods=['POST'])
def result():
    if request.method == 'POST':
            if 'file' not in request.files:
            return render_template('index.html', error='No  file  part')
        file = request.files['file']
            if file.filename == '':
            return render_template('index.html', error='No selected file')
        if file:
            # Save the uploaded file
            filename = secure_filename(file.filename)
            file_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
            file.save(file_path)
            # Process the image
            result = detect_disease(file_path)
            return render_template('result.html', result=result)
if _name_ == '_main_':
    app.run(debug=True)
```

#ml code#

```
import os, torch, shutil, numpy as np
from glob import glob; from PIL import Image
from torch.utils.data import random_split, Dataset, DataLoader
from torchvision import transforms as T
torch.manual_seed(2024)
class CustomDataset(Dataset):
    def__init__(self, root, transformations = None):
        self.transformations = transformations
        self.im_paths = [im_path for im_path in sorted(glob(f"{root}/*/*"))]
        self.cls_names, self.cls_counts, count, data_count = {}, {}, 0, 0
        for idx, im_path in enumerate(self.im_paths):
            class_name = self.get_class(im_path)
            if class_name not in self.cls_names: self.cls_names[class_name] =
count; self.cls_counts[class_name] = 1; count += 1
            else: self.cls_counts[class_name]  += 1
    def get_class(self, path): return os.path.dirname(path).split("/")[-1]
    def__len__(self): return len(self.im_paths)
    def__getitem__(self, idx):
```

```
        im_path = self.im_paths[idx]
        im =  Image.open(im_path).convert("RGB")
        gt  = self.cls_names[self.get_class(im_path)]
        if self.transformations is not None: im = self.transformations(im)
        return im, gt
def get_dls(root, transformations, bs, split = [0.9, 0.05, 0.05], ns = 4):
    ds = CustomDataset(root = root, transformations = transformations)
    total_len = len(ds)
    tr_len = int(total_len * split[0])
    vl_len = int(total_len * split[1])
    ts_len = total_len - (tr_len + vl_len)
    tr_ds, vl_ds, ts_ds = random_split(dataset = ds, lengths = [tr_len, vl_len,
ts_len])
    tr_dl, val_dl, ts_dl = DataLoader(tr_ds, batch_size = bs, shuffle = True,
num_workers = ns), DataLoader(vl_ds, batch_size = bs, shuffle = False,
num_workers = ns), DataLoader(ts_ds, batch_size = 1, shuffle = False,
num_workers = ns)
    return tr_dl, val_dl, ts_dl, ds.cls_names
root = "/kaggle/input/sugarcane-leaf-disease-dataset"
mean, std, im_size = [0.485, 0.456, 0.406], [0.229, 0.224, 0.225], 224
tfs = T.Compose([T.Resize((im_size, im_size)), T.ToTensor(),
T.Normalize(mean = mean, std = std)])
tr_dl, val_dl, ts_dl, classes = get_dls(root = root, transformations = tfs, bs = 16)
print(len(tr_dl)); print(len(val_dl)); print(len(ts_dl)); print(classes)
import random

from matplotlib import pyplot as plt

def tensor_2_im(t, t_type = "rgb"):

        gray_tfs = T.Compose([T.Normalize(mean = [ 0.], std = [1/0.5]),
T.Normalize(mean = [-0.5], std = [1])])

    rgb_tfs = T.Compose([T.Normalize(mean = [ 0., 0., 0. ], std = [ 1/0.229,
1/0.224, 1/0.225 ]), T.Normalize(mean = [ -0.485, -0.456, -0.406 ], std = [ 1., 1.,
1. ])])



    invTrans = gray_tfs if t_type == "gray" else rgb_tfs

    return (invTrans(t) *
255).detach().squeeze().cpu().permute(1,2,0).numpy().astype(np.uint8)  if t_type
== "gray" else (invTrans(t) *
255).detach().cpu().permute(1,2,0).numpy().astype(np.uint8)

def visualize(data, n_ims, rows, cmap = None, cls_names = None):
```

```
    assert cmap in ["rgb", "gray"], "Rasmni oq-qora yoki rangli ekanini aniqlashtirib bering!"
    if cmap == "rgb": cmap = "viridis"
    plt.figure(figsize = (20, 10))
    indekslar = [random.randint(0, len(data) - 1) for _ in range(n_ims)]
    for idx, indeks in enumerate(indekslar):
        im, gt = data[indeks]
        # Start plot
        plt.subplot(rows, n_ims // rows, idx + 1)
        if cmap: plt.imshow(tensor_2_im(im, cmap), cmap=cmap)
        else: plt.imshow(tensor_2_im(im))
        plt.axis('off')
        if cls_names is not None: plt.title(f"GT -> {cls_names[int(gt)]}")
        else: plt.title(f"GT -> {gt}")
  visualize(tr_dl.dataset, 20, 4, "rgb", list(classes.keys()))
import timm
from tqdm import tqdm
m = timm.create_model("rexnet_150", pretrained = True, num_classes = len(classes))
def train_setup(m): return m.to("cpu").eval(), 20, "cpu", torch.nn.CrossEntropyLoss(), torch.optim.Adam(params = m.parameters(), lr = 3e-4)
def to_device(batch, device): return batch[0].to(device), batch[1].to(device)
def get_metrics(model, ims, gts, loss_fn, epoch_loss, epoch_acc): preds = model(ims); loss = loss_fn(preds, gts); return loss, epoch_loss + (loss.item()), epoch_acc + (torch.argmax(preds, dim = 1) == gts).sum().item()

m, epochs, device, loss_fn, optimizer = train_setup(m)
save_prefix, save_dir = "sugarcane", "saved_models"
print("Start training...")
```

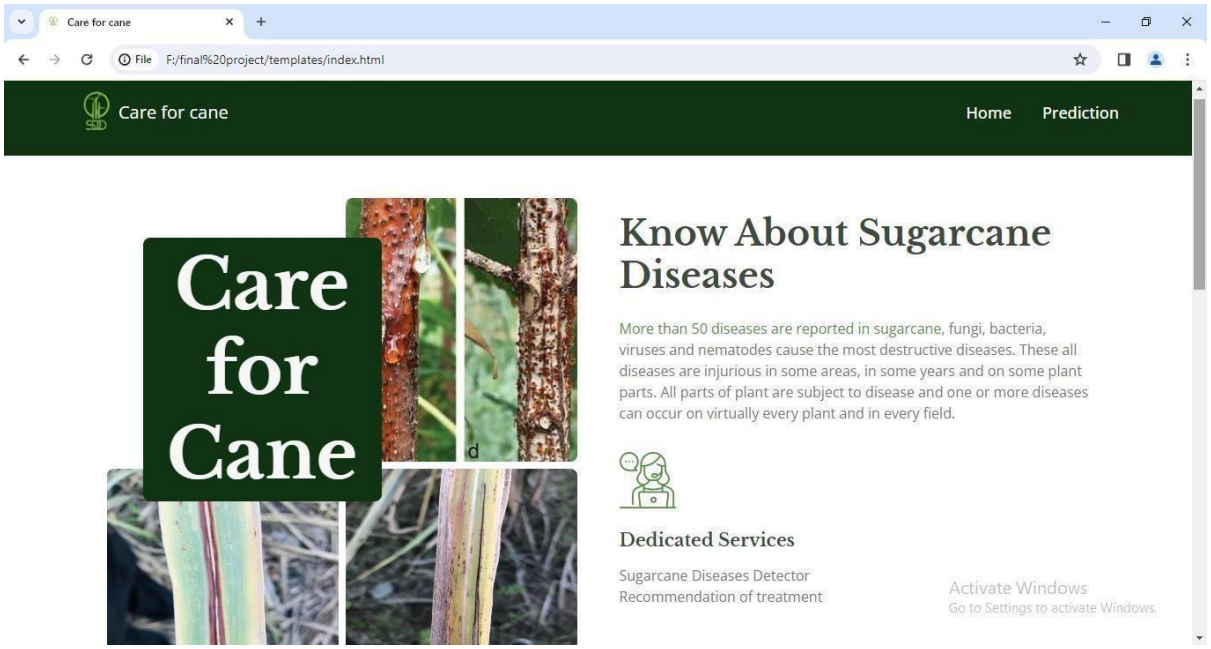
```
best_acc, best_loss, threshold, not_improved, patience = 0, float("inf"), 0.01, 0,
5
tr_losses, val_losses, tr_accs, val_accs = [], [], [], []
best_loss = float(torch.inf)
for epoch in range(epochs):
    epoch_loss, epoch_acc = 0, 0
    for idx, batch in tqdm(enumerate(tr_dl)):
        ims, gts = to_device(batch, device)
        loss, epoch_loss, epoch_acc = get_metrics(m, ims, gts, loss_fn,
epoch_loss, epoch_acc)
        optimizer.zero_grad(); loss.backward(); optimizer.step()
    tr_loss_to_track = epoch_loss / len(tr_dl)
    tr_acc_to_track = epoch_acc / len(tr_dl.dataset)
    tr_losses.append(tr_loss_to_track); tr_accs.append(tr_acc_to_track)
    print(f"{epoch + 1}-epoch train process is completed!")
    print(f"{epoch + 1}-epoch train loss          -> {tr_loss_to_track:.3f}")
    print(f"{epoch + 1}-epoch train accuracy      -> {tr_acc_to_track:.3f}")
    m.eval()
    with torch.no_grad():
        val_epoch_loss, val_epoch_acc = 0, 0
        for idx, batch in enumerate(val_dl):
            ims, gts = batch
            ims, gts = ims.to(device), gts.to(device)
            preds = m(ims)
            loss = loss_fn(preds, gts)
            pred_cls = torch.argmax(preds.data, dim = 1)
            val_epoch_acc += (pred_cls == gts).sum().item()
            val_epoch_loss += loss.item()
        val_loss_to_track = val_epoch_loss / len(val_dl)
```

```
        val_acc_to_track = val_epoch_acc / len(val_dl.dataset)
        val_losses.append(val_loss_to_track); val_accs.append(val_acc_to_track)
        print(f"{epoch + 1}-epoch validation process is completed!")
        print(f"{epoch + 1}-epoch validation loss      -> {val_loss_to_track:.3f}")
        print(f"{epoch + 1}-epoch validation  accuracy  ->  {val_acc_to_track:.3f}")
        if val_loss_to_track < (best_loss + threshold):
            os.makedirs(save_dir, exist_ok = True)
            best_loss = val_loss_to_track
            torch.save(m.state_dict(), f"{save_dir}/{save_prefix}_best_model.pth")
        else:
            not_improved += 1
            print(f"Loss value did not decrease for {not_improved} epochs")
            if not_improved == patience:
                print(f"Stop training since loss value did not decrease for {patience}
epochs.")
                break
import cv2
class SaveFeatures():
    def  getCAM(conv_fs,  linear_weights,  class_idx):
     bs, chs, h, w = conv_fs.shape
    cam = linear_weights[class_idx].dot(conv_fs[0,:, :, ].reshape((chs, h * w)))
    cam = cam.reshape(h,  w)
    return (cam - np.min(cam)) /  np.max(cam)
def inference(model, device, test_dl, num_ims, row, final_conv, fc_params,
cls_names = None):
       weight, acc = np.squeeze(fc_params[0].cpu().data.numpy()), 0
    activated_features = SaveFeatures(final_conv)
    preds, images, lbls = [], [], []
    for  idx,  batch  in  tqdm(enumerate(test_dl)):
```

```
        im, gt = to_device(batch, device)
        pred_class = torch.argmax(model(im), dim = 1)
        acc += (pred_class == gt).sum().item()
        images.append(im)
        preds.append(pred_class.item())
        lbls.append(gt.item()
    print(f"Accuracy of the model on the test data -> {(acc / len(test_dl.dataset)):.3f}")
    plt.figure(figsize = (20, 10))
    indekslar = [random.randint(0, len(images) - 1) for _ in range(num_ims)]
    for idx, indeks in enumerate(indekslar):
        im = images[indeks].squeeze()
        pred_idx = preds[indeks]
        heatmap = getCAM(activated_features.features, weight, pred_idx)
        # Start plot
        plt.subplot(row, num_ims // row, idx + 1)
        plt.imshow(tensor_2_im(im), cmap = "gray"); plt.axis("off")
        plt.imshow(cv2.resize(heatmap, (im_size, im_size), interpolation=cv2.INTER_LINEAR), alpha=0.4, cmap='jet'); plt.axis("off")
        if cls_names is not None: plt.title(f"GT -> {cls_names[int(lbls[indeks])]} ; PRED -> {cls_names[int(preds[indeks])]}", color=("green" if {cls_names[int(lbls[indeks])]} == {cls_names[int(preds[indeks])]} else "red"))
        else: plt.title(f"GT -> {gt} ; PRED -> {pred}")
m.load_state_dict(torch.load(f"{save_dir}/{save_prefix}_best_model.pth"))
m.eval()
final_conv, fc_params = m.features[-1], list(m.head.fc.parameters())
inference(model = m.to(device), device = device, test_dl = ts_dl, num_ims = 20, row = 4, cls_names = list(classes.keys()), final_conv = final_conv, fc_params = fc_params)
```

9.2 SCREENSHOTS

OUTPUT



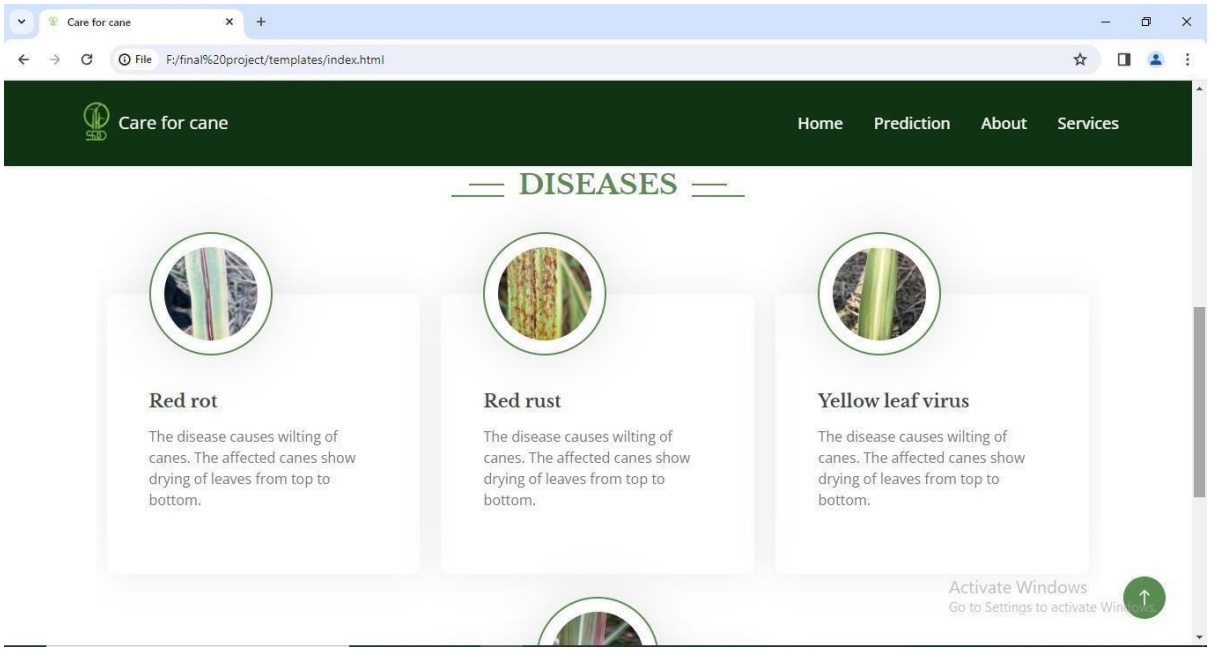


Fig.9.2.1 HOME PAGE

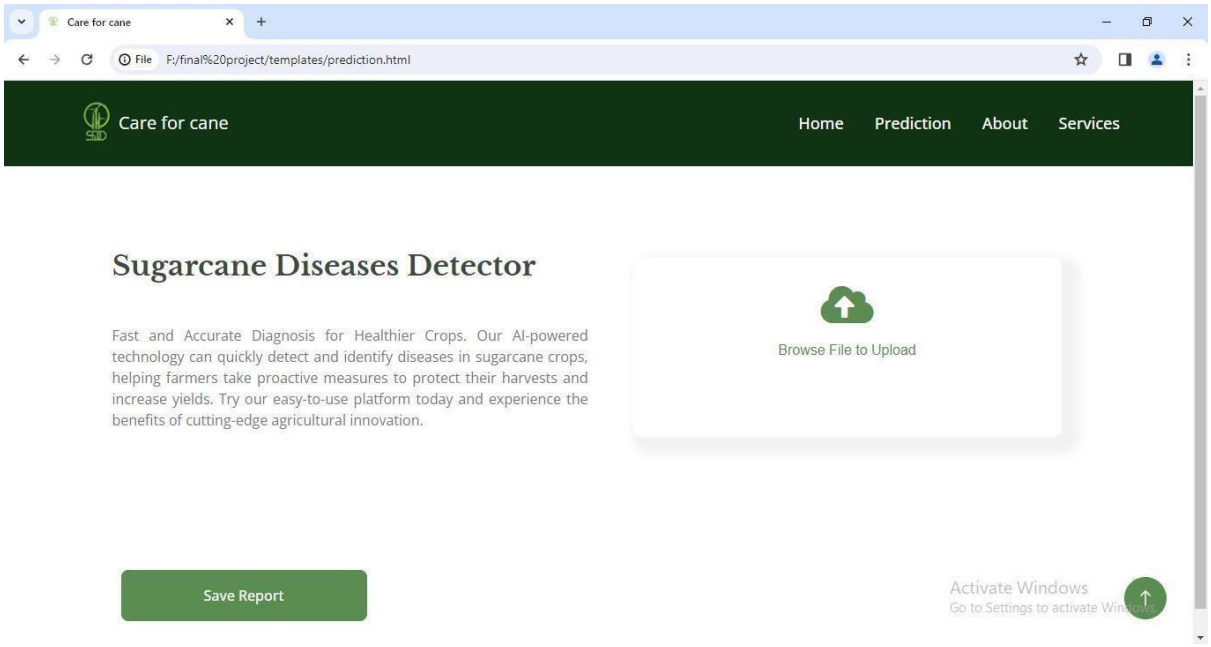


Fig.9.2.2 BROWSE THE FILE

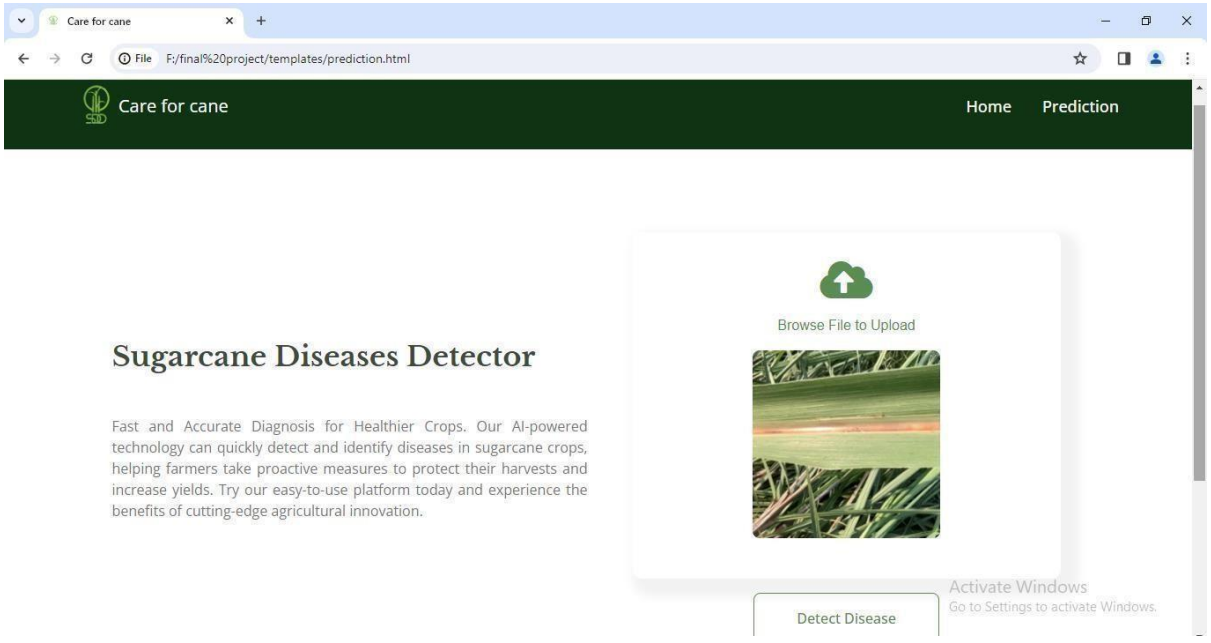


Fig.9.2.3 IMAGE UPLOAD

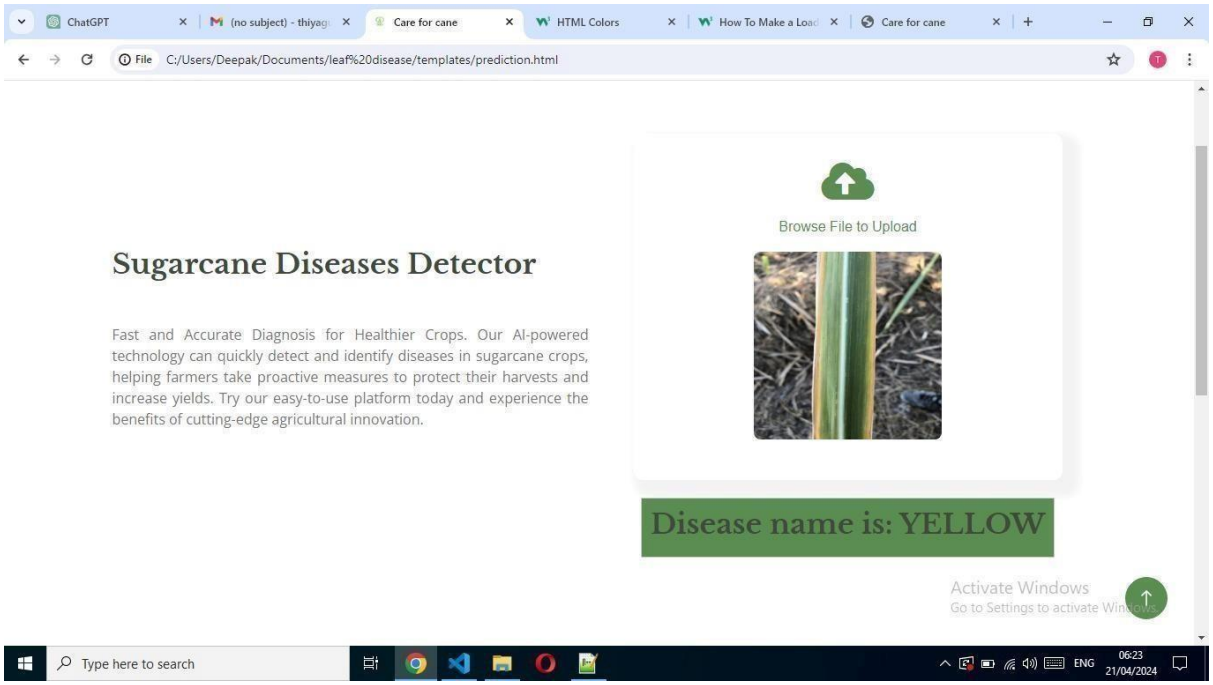


Fig.9.2.4 PREDICTED RESULT

Fig.9.2.5 VISUALIZE DATASET

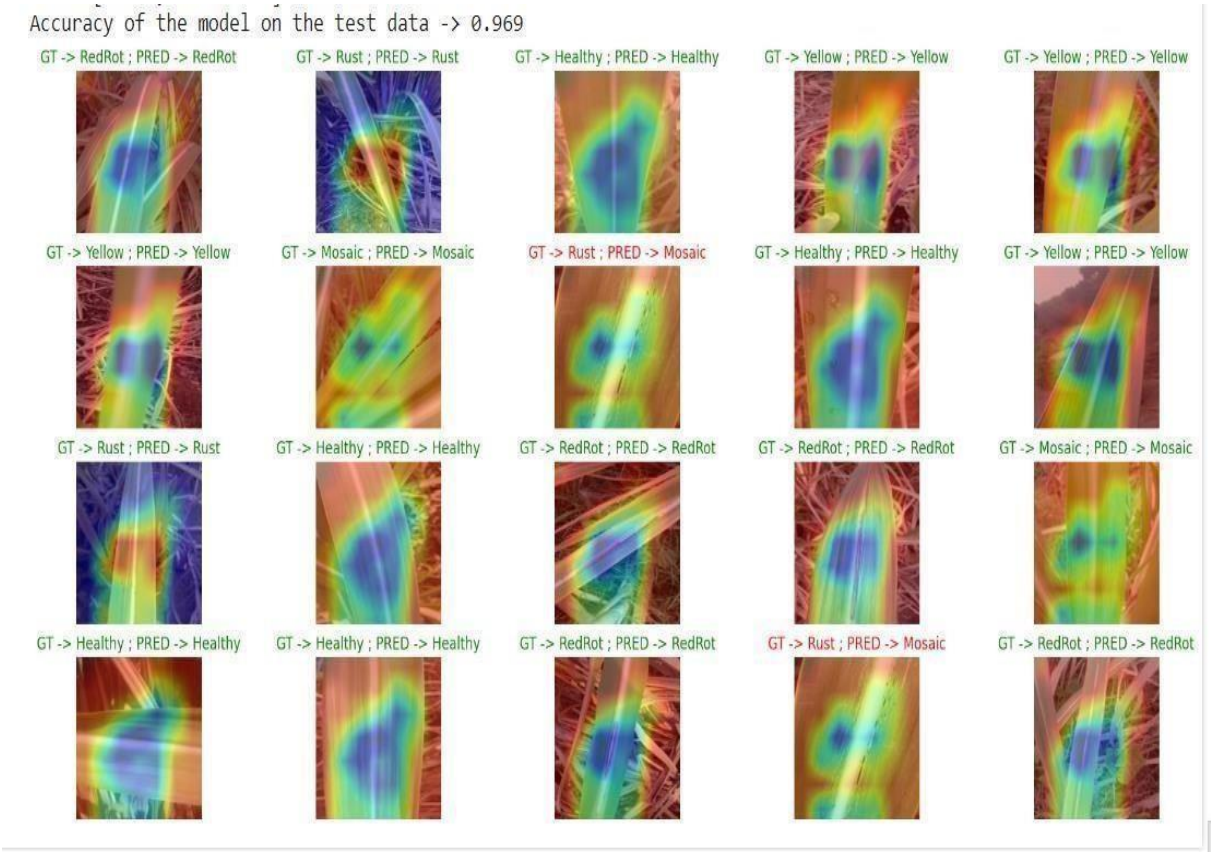


FIG 9.2.6 PREDICTION OF DISEASES

CHAPTER - 10
REFERENCES

[1] Laskmikanth Paleti, Arava Nagasri and Sunitha (2023) 'Sugarcane leaf disease classification and identification using deep learning algorithms', Journal of Theoretical and Applied Information Technology (JATIT), Vol.101, No.20 pp.6460 - 6472.

[2] Sammy V. Militante, Bobby D. Gerardo, Ruji P. Medina (2019) 'Sugarcane disease recognition using deep learning', Institute of Electrical and Electronics Engineers Eurasia Conference on IOT, Communication and Engineering (IEEE) Vol.20, pp.575 – 578.

[3] Santhrupth B. C., Devaraj Verma C (2023) 'Intelligent disease detection in sugarcane plants', Intelligent Systems and Applications in Engineering (IJISAE), Vol.12, pp.299 – 306.

[4] Anish Kuckian, Yash Kankriya (2022) 'Sugarcane disease detection using deep learning', International Conference on Advance in Science and Technology(ICAST), Vol.22, pp.40 - 44.

[5] Snehal Pawar, Jyotismita Talukdar (2019) 'Sugarcane leaf disease detection', International Research Journal of Engineering and Technology (IRJET), Vol.06, pp.3025 – 3028.

[6] H. Park, S. H. Kim, (2017) 'Image – based disease diagnosing and predictingof the crops through crops through the deep learning mechanisms', Information and Communication Technology Convergence (ICTC) IEEE 2017 International Conference, Vol.4, No.12, pp.129 - 131.

[7] K. Elangovan and S. Nalini,(2018) ‘Plant disease classification using image segmentation and SVM techniques’, International Journal of Computational Intelligence Research (IJCIR), Vol. 13, No.7, pp.1821- 1828.

[8] C. Hortinela et.al., (2019) ‘Classification of cane sugar based on physical characteristics using SVM’,2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Communication and Control, Environment, and Management (HNICEM), Vol.9, No.12, pp.1 – 5.

[9] Arti. N. Rathod et al,(2013) ‘Image processing techniques for detection of leaf disease’, International Journal of Advanced Research in Computer Science and Software Engineering (IJARCS