**PADDY DISEASES RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS**

**Abstract:**

Objective of this work is focused on developing a system capable of identifying various diseases of paddy plant. Faster RCNN, a model of Convolutional Neural Network (CNN) has been executed for recognition. TensorFlow library has been used in Anaconda environment. Samples of diseased plants were collected from Bangladesh Rice Research Institute (BRRI), Rajshahi. Before initiating training procedure, the data images had to be processed. Then after completion of training program, the recognition was possible automatically. A design of a real time system has also been proposed.

**Overview of the project**

This work presents a CNN based model which **provides 97.40% accurate results in predicting various diseases of rice leaves**. Using a dataset of over 900 images of diseases and healthy leaves and following the technique of 10-fold cross validation, the model was trained to identify 4 common rice diseases.

Objectives

In the Indian economic sector, agriculture plays an important role and contributes a second place in rice production. Approximately all the states in India practicing rice cultivation which are Tamil Nadu, West Bengal, Punjab, Uttar Pradesh, Assam, Bihar, etc. Agriculture sector is providing its share about 19.9% to the total gross domestic product. Rice is one of the predominantly using food grains in India. The growth and quality of rice plants get affecting by diseases which further implies the profit of the farming. The different varieties of diseases may occur in individual rice crop which is difficult to identify to the farmers with their limited knowledge gained through the experience. For this precision and early identification of plant diseases diagnosis automatic data processing expert system is crucially requisite. Thus the healthy and successful cultivation is viable.

Scope of the project

In this study, a pre-trained InceptionResNetV2 model incorporates the knowledge in the form of weights which is transformed to our experiment for the feature extraction process by using transfer leaning approach. The deep learning model has been fine-tuned for getting the improved accuracy in identification of the various classes of rice leaf diseases

**ABSTRACT**

A variety of diseases attack rice, one of India's most widely planted crops, at various phases in the growing process. It is exceedingly difficult for farmers with poor understanding to identify these diseases manually. Automated picture identification systems based on Convolutional Neural Network (CNN) models are showing considerable promise in deep learning research recently. Our deep learning model was created using Transfer Learning on a small dataset because it was difficult to find an image dataset of rice leaf disease. VGG-16 is utilized to train and assess the proposed CNN architecture, which is based on rice field and internet datasets. The proposed model has a 95 percent accuracy rate. Deep Learning, Convolutional Neural Network (CNN), fine-tuning and rice leaf diseases are some of the terms in this index.

**1.INTRODUCTION**

In India and around the world, rice is the primary source of nutrition. Throughout the course of its development, it is plagued by a wide range of ailments. A high-quality crop depends on early diagnosis and treatment of these diseases, but this is challenging due to the enormous tracts of land controlled by individual farmers, the great variety of diseases they carry, and the likelihood of many diseases in a single crop. It's tough and time-consuming to locate agricultural experts in remote areas. As a result, Automated Systems are required. To help farmers in need and improve the precision with which plant illnesses can be identified, artificial neural networks (ANNs) and support vector machines (SVMs) have been utilised in studies. But the accuracy of these systems depends greatly on the features that are selected. Images may now be recognised using convolutional neural networks, which eliminates the requirement for picture preprocessing and provides built-in feature selection. Another issue is that huge datasets for these kinds of challenges are extremely difficult to come by. In situations where the dataset size is relatively small, it is preferable to use a model that has been trained on a large dataset. When developing a new model using Transfer Learning, it is possible to delete the final layer of connections or to fine tune the final layers to be more specific to the dataset under consideration. Farmers can send photographs of infected leaves taken using their smartphones to our server, where our neural network will identify the illness and return to the farmers with a diagnosis and treatment recommendations. This notion came to us as a result of the widespread availability of cell phones. A disease classification component for an automated system is presented in this paper. Convolutional neural network work has helped us build a deep learning technique in this work... Transfer Learning has been utilised to fine-tune the fully connected layers so that we can fit our own datasets into the VGG-16 model's fully connected layers. At the end, we looked at our mistakes and tried to figure out why they happened.

**METHODOLOGY**

The architecture of convolutional neural networks (CNNs) dictates the network's performance. The convolution layer, the pooling layer, and the fully linked layer are all part of it. Three layers are involved in this process: a feature extractor (the first two) and a classification layer (the third). Convolutional features are reduced in dimension thanks to the pooling layer. The feature extracted by the fully connected layer is then used by softmax to classify the images. The learnable filters in the convolution layer retrieve the information from the input image. The two-dimensional feature map is created by taking the dot product of each filter with the raw picture pixel in a sliding window. The ReLU is one of the most common activation functions. Feature maps can be reduced by using "max pooling," a subsampling layer. It's then possible to connect each feature map to the completely connected layer. Using Softmax, each class is assigned a decimal probability. Pre-trained weights from the ImageNet Database are used to train VGG-16's architecture, which has access to 3.2 million photos annotated in great detail across 5247 possible categories in total. By using pre-trained models, it is possible to move weights, architecture, and features from one domain to another. There are more data-specific features in the upper layers. The last dense layer with a size of 128 is used for classification, followed by a softmax layer with a size of 4. (because there are four classes). In comparison to prior techniques, CNN is a feature extractor, which gives it an edge over CNN. We use our dataset to retrain the VGG-16 model and fine-tune it in order to get the classifications.

Given the size and familiarity of the new dataset, four transfer learning procedures are possible:

Small and familiar, the new dataset is derived from the old.

Unlike the previous dataset, the new dataset is both huge and familiar.

Even though it's a minor addition, the new data is quite distinct from the old.

It's a massive change from the last one, and it's completely new. It follows that in order to employ the VGGNet as a feature extractor until the final fully connected layers that were fine-tuned in order to meet the number of classes in our dataset, our model falls into the third situation as outlined above. Details of the proposed architecture are displayed. Each of the four convolution layers is followed by a ReLU, Maxpooling, and dropout layer, and then two Fully Connected Layers (FCL) and SoftMax are added on top of that. Although there was a substantial difference in performance, it was still evident. A comparison of the two is included in the findings section.

**2.LITERATURE SURVEY**

* 1. **Using Deep Learning for Image-Based Plant Disease Detection**

**AUTHORS:** V. Singh, A. Misra

**ABSTRACT:** Lack of infrastructure makes it difficult to detect agricultural diseases rapidly, despite their considerable threat to global food security. Because of the extensive usage of smartphones around the world and recent improvements in computer vision made possible by deep learning, disease diagnosis by smartphone is becoming a reality. We train a convolutional neural network to recognise 14 crop species and 26 illnesses from a publically available dataset using 54,306 pictures of damaged and healthy plant leaves (or absence thereof). Using a held-out test set, the trained model achieves an accuracy of 99.35%, demonstrating the validity of the approach. Using a set of shots that were not taken in the same conditions as those used for training, the model's accuracy is just 31.4 percent The total accuracy can still be improved by using a more diversified set of training data, though. For crop disease diagnosis on a global scale, training deep learning models on increasingly large and publically available image datasets provides a clear path.

**2.2SVM classifier based grape leaf disease detection**

**AUTHORS:** P. B. Padol and A. A. Yadav

**ABSTRACT:** Grapes are one of India's most popular fruit crops. Grape yields fall as a result of disease infections on the fruit, stems, and leaves. Toxins from bacteria, fungus or virus are the most common causes. There are many factors that limit the amount of fruit that can be produced, including illness. It's impossible to implement effective control measures without an accurate diagnosis of the disease. For the identification and categorization of plant leaf diseases, image processing is a commonly employed technique. SVM classification is used in this work to aid in the detection and categorization of grape leaf diseases. Segmentation by K-means clustering is used to locate the sick area, and then colour and texture features are extracted. Leaf disease can finally be identified using a classification system. Accuracy in the suggested approach is 88.89 percent for the condition being studied.

**2.3 Very Deep Convolutional Networks for Large Scale Image Recognition**

**AUTHORS: Karen Simonyan, Andrew Zisserman**

**ABSTRACT:** Using a large-scale image identification task, we investigate how a convolutional network's depth influences its performance. Using an architecture with extremely small (3x3) convolution filters, we examined networks of increasing depth and found that raising the depth to 16-19 weight layers significantly improved performance over prior-art setups. This is the main contribution of our study. As a result of these discoveries, our team was able to take first and second place in the 2014 ImageNet Challenge in the categories of localization and classification. A wide range of datasets can benefit from our representations' state-of-the-art results. Two of our finest ConvNet models have been provided so that further research into deep visual representations in computer vision can take place.

# 2.4Hyper-class augmented and regularized deep learning for fine-grained image classification

**AUTHORS:** S. Xie, T. Yang, X. Wang, and Y. Lin

**ABSTRACT:** Deep convolutional neural networks have shown significant success in large-scale object recognition (CNNs). Due to the high cost of fine-grained labelled data (usually necessitating domain expertise), as well as the huge intra-class and moderate inter-class variance, fine-grained image classification (FGIC) is significantly more challenging than generic object identification. Using an external dataset (like ImageNet) to pre-train the CNN and then fine-tuning it on the small target data to meet a specific classification goal is one of the most prevalent ways. Two new aspects of the problem of learning a deep CNN are introduced in this paper: identifying easily annotated hyper-classes inherent in the fine-grained data and collecting numerous images with hyper-classes labelled from readily available external sources (such as image search engines), formulating the problem as multitask learning; and (iii) a novel learning modality, which is based on the idea of a "multitask learning" paradigm. Two small, fine-grained datasets (Stanford Dogs and Stanford Cars) as well as a big, automotive-specific dataset have all been used to evaluate the proposed approach.

**2.5 Detection of Plant Leaf Diseases Using Image Segmentation and Soft Computing Techniques**

**AUTHORS:** V. Singh, A. Misra,

**ABSTRACT:**The economy heavily relies on agricultural productivity. It's for this reason, as well as the fact that diseases in plants are relatively common, that disease detection in the agricultural sector is so crucial. Neglecting this region can have detrimental impacts on plants, which in turn impair the quality, quantity, and productivity of the final output. For example, in the United States, pine trees are susceptible to a dangerous illness known as small leaf disease. The early detection of disease symptoms, such as those that occur on plant leaves, is an advantage of using an automated system for detecting plant disease. This method minimises the amount of time needed to monitor huge farms of crops. For the automatic identification and classification of plant leaf diseases, this research provides an algorithm for the picture segmentation technique. A review of various disease classification strategies for plant leaf disease detection is also included. Genetic algorithm is used for image segmentation, which is critical for disease identification in plant leaf disease.

**3.SYSTEM ANALYSIS**

**3.1 EXISTING SYSTEM:**

In the past, the only way to identify a disease was to manually examine the leaf. The illness was discovered using a combination of visual inspection of plant leaves and consulting a reference book. This method has three key drawbacks: limited accuracy, the inability to analyse every leaf, and a lengthy time investment. As science and technology progress, new methods for accurately diagnosing these diseases emerge. Image processing and deep learning are two methods to consider. Filtering, clustering and histogram analysis are some of the approaches used in image processing to identify the diseased area. Deep learning neural networks, on the other hand, are used to detect disorders.

**3.1.1 DISADVANTAGES OF EXISTING SYSTEM:**

* It's impossible to study every leaf, and it takes a long time.
  1. **PROPOSED SYSTEM:**

A VGG16 transfer learning neural network is used in this paper to train a dataset of rice diseases, and the trained model may be used to predict disease from new photos. Because the Rice Leaf dataset from KAGGLE was too tiny for author to train the VGG16 model, he turned to the transfer learning CNN algorithm, which transfers an existing CNN model to a new dataset and then uses the new data to train the model.

It has been shown that VGG16 transfer learning improves prediction accuracy in both a normal CNN model and a normal CNN model with VGG16 transfer learning.

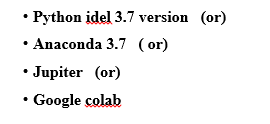
**3.2.1 ADVANTAGES OF PROPOSED SYSTEM:**

* • VGG16 can be utilised to identify rice leaf diseases. Over ninety-five percent of the time, it has been right.

**3.3 SYSTEM REQUIREMENTS:**

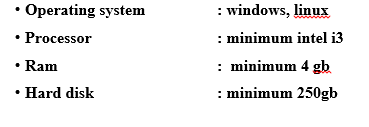
**SOFTWARE REQUIREMENTS**

Product viewpoint and features, OS and operating environment, graphics requirements, design limitations and user documentation are all included in the functional requirements or overall description documents.

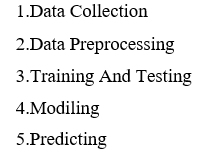


**HARDWARE REQUIREMENTS**

Enthought Python / Canopy / VS Code users' hardware requirements are highly dependant on the software they are developing. Large arrays/objects in memory demand more RAM, but apps that need to do several calculations or jobs quickly require a faster processor.



### **FUNCTIONALREQUIREMENTS**



**3.5 NON FUNCTIONALREQUIREMENTS**

* A software system's quality trait is known as NON-FUNCTIONAL REQUIREMENT (NFR). These non-functional criteria are crucial to the success of the software system and are judged on their responsiveness, usability, security, and portability. "How quickly does the website load?" is an example of a nonfunctional demand. Systems that fall short of user expectations may be the result of a failure to meet non-functional requirements. Through the use of non-functional requirements, it is possible to place limitations on the system's design in various agile backlogs. With a concurrent user count of more than 10,000, the site should be able to load in three seconds. Functional and non-functional criteria are equally important.
* a requirement for ease of use; a requirement for ease of maintenance;
* • The need for manageability
* In order to meet the recovery criteria,
* • The need for security
* To meet the Data Integrity standard,
* • Demand for capacity
* • Requirement of availability
* • Need for scalability
* • The need for interoperability
* Reliability is essential.
* • The demand for upkeep
* Legislative mandate
* In terms of the environment, this is an important consideration.

**3.6 SYSTEM STUDY**

**FEASIBILITY STUDY**

Project feasibility and cost estimates are calculated at this point, and a business proposal is drafted to go along with it. The proposed system's feasibility will be studied during system analysis. As a precautionary measure, There will be no additional costs to the company as a result of the proposed system. Feasibility studies necessitate a grasp of the system's fundamental requirements.

Three key components make up the feasibility analysis.

FLEXIBILITY OF THE MARKET

In terms of technical feasibility, this is the most important factor.

FLEXIBILITY IN SOCIAL CONTEXT

COST-BENEFITABILITY

An evaluation of the system's financial impact on the organisation is the purpose of this research.

FELIABILITY IN TECHNICAL TERMS

The purpose of this investigation is to see if the system's technical criteria can be met. A system's technological resources shouldn't be taxed beyond their capacity when it is under development. As a result, the limited technical resources will be put under a lot of stress. This will put a lot of pressure on the customer. The designed system must have a low demand because only little or no adjustments are needed to implement this system.

SOCIAL COMFORT

The study's main objective is to determine how well the system is perceived by the intended audience it is designed for. Educating the user on how to get the most out of the technology falls under this category. The system should not be viewed as a threat, but rather as a need. The methods employed to educate and acclimate the user to the system have the exclusive influence on the level of user acceptance. In order for him to be able to provide some useful input as the system's ultimate consumer, his self-esteem must be boosted.

**RELATED WORK**

Many studies have used classical classifiers, however the outcomes are dependent on the feature selection approaches and picture preprocessing is a significant stage in the process of study. There are numerous studies that are taking use of CNN's high recognition accuracy because of that.

A. Detection of Plant Diseases using CNN Paper-based training uses 87,848 photos from 25 plant species and the 58 groups they fall under, including examples of "healthy" plant growth. One of the best models has a 99.53% success rate in correctly classifying objects. 54306 photos of 14 different crop kinds, each with 26 different illnesses and healthy leaves, were used to train CNN. When evaluated on a different dataset derived from a real-world event, the 99.35 percent success rate dropped to 31.4%. We'll talk about why determining the severity of a condition is more difficult than just classifying it. Because of the high levels of intra-class resemblance across photos belonging to the same class, it is even more difficult to identify it.

The use of CNN to detect rice disease

A classifier based on a convolutional neural network was trained on 227 photos of healthy and damaged rice plants.

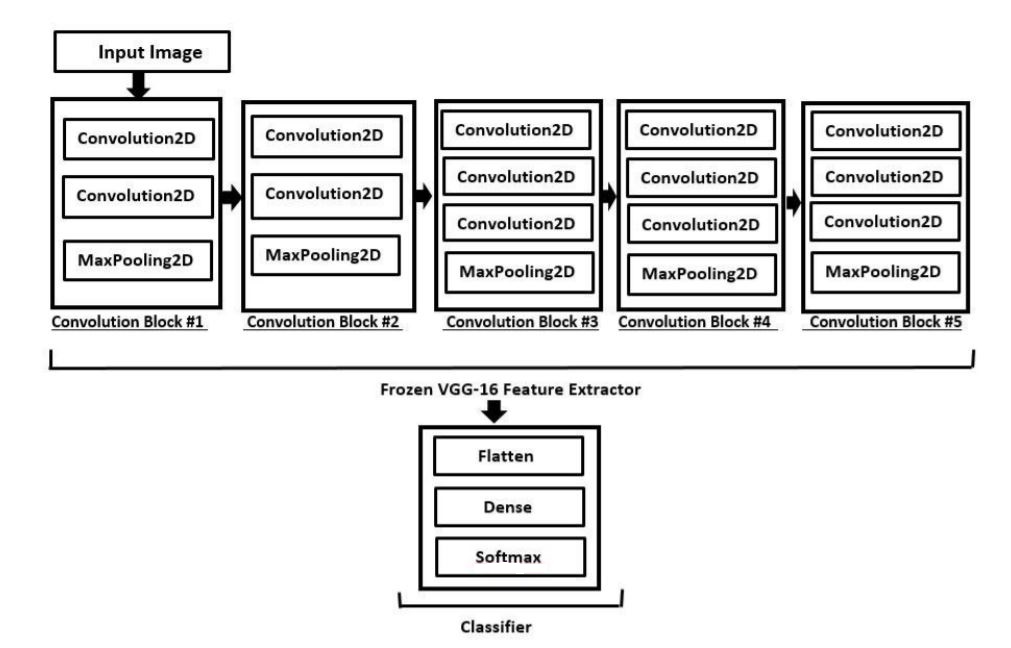
AlexNet is used to train the classifier. 91.23% of the time, the aforementioned architecture is able to accurately forecast whether or not a plant is unhealthy. The authors gathered 500 photos of ten distinct leaf and stem rice illnesses. Le-Net and AlexNet were used as inspiration for the design of its architecture, which was tested and achieved 95.48% accuracy. Pre-processing steps including image resizing to 512\*512, normalisation, PCA and whitening were performed because the data was so sparse. Instead of using max pooling, they opted for stochastic pooling, which they claimed prevented overfitting.

**4.SYSTEM DESIGN**

**4.1 SYSTEM ARCHITECTURE:**

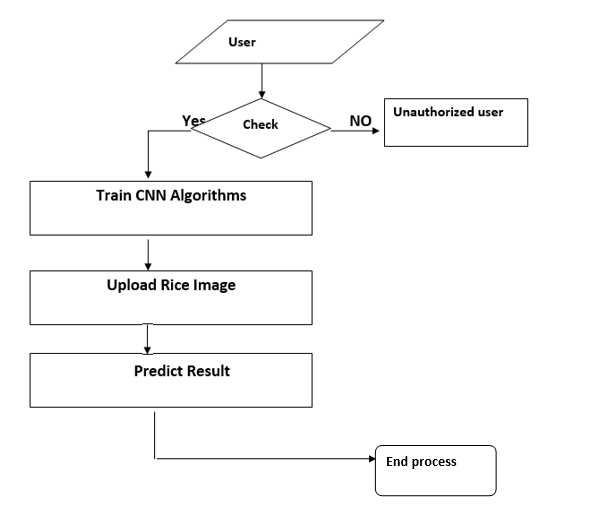
**4.1 SYSTEM ARCHITECTURE:**

For convolutional neural networks, the VGG16 CNN design is used. It remains one of the greatest vision model designs to this day. Instead of a slew of hyper-parameters, the 3x3 filter's convolution layers with stride 1 and the 2x2 filter's padding and maxpool layer with stride 2 were utilised for VGG16's most distinctive feature. The convolution and max pool layers are placed in the same way throughout the architecture. Two FC (fully connected layers) and a softmax are used for output. VGG16 has 16 weighted layers, as indicated by the number "16" in its name.. This network contains approximately 138 million variables.

****

**4.2DATA FLOW DIAGRAM:**

1. The DFD can also be referred to as bubble charts. For example, a simple visual formalism can portray a system in terms of the input and output of data.
2. Second, the data flow diagram is a crucial modelling tool (DFD). Models the system's components. System processes, data, an external entity interacting with the system, and information flow within the system are all included in this list.
3. The DFD displays how information flows through the system and how it is transformed through a sequence of transformations. It's a visual representation of how data goes from input to output and the transformations that take place along the way.
4. 4. DFD, or bubble chart, is a type of graph. A DFD can be used at any degree of abstraction to represent a given system. To reflect increasing information flow and functional detail, DFD might be partitioned into a number of tiers or layers.



**4.3 UML DIAGRAMS**

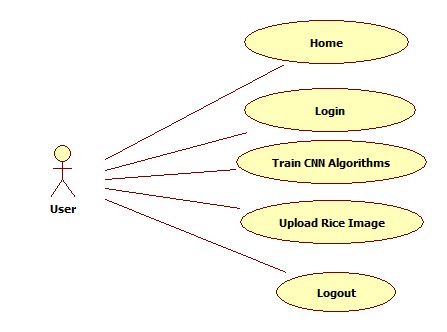
1. UML stands for Unified Modeling Language. UML is a general-purpose modelling language used in object-oriented software engineering. The standard was established by the Object Management Group.
2. Second, it is envisaged that the widespread use of object-oriented computer software models will result from the extensive use of UML. UML's present structure is mostly composed of a Meta-model and a Notation. Other methods and processes may be added in the near future to UML as well. Software system artefacts and other non-software system models can be specified, visualised, built and documented in a single vocabulary known as the Unified Modeling Language (UML) (UML).
3. To build large, complex systems, the UML use a variety of tried-and-true engineering approaches.
4. The UML is an essential part of object-oriented software development and the software development process. The UML is widely used to represent software project design in graphical notations.

GOALS:

1. The following are the primary aims of the UML design:
2. Provide users with an expressive visual modelling Language so that they may create and trade meaningful models.
3. Provide means for extending the fundamental concepts through extendibility and specialisation.
4. Programming languages and development processes should be able to stand on their own two feet.
5. A formal framework for understanding the modelling language is provided in this step.
6. Assist the expansion of the market for object-oriented (OO) tools.
7. Assist with the implementation of higher-level development concepts such as teamwork and frameworks.
8. Best practises should be incorporated.

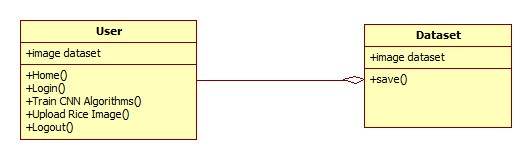
**Use case diagram:**

Diagrams in the Unified Modeling Language (UML) based on use cases are known as use cases in the UML. Use cases, actors, and any dependencies between them are all depicted graphically to give a clear picture of the system's capabilities. A use case diagram's primary goal is to explain how the system performs for each actor. System actors can be depicted by their responsibilities in the system.

****

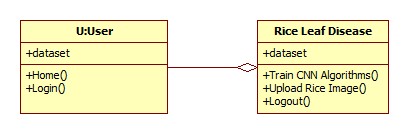
**Class diagram:**

Using the class diagram, you may fine-tune the use-case diagram and create a more precise blueprint for your system's architecture. As seen in the use case diagram, each actor is assigned to one of several classes, which are themselves subsets of the larger class diagram. Classes can have a "is-a" or a "has-a" relationship, depending on the context. Each class depicted in the diagram may be able to do a specific task. The class's "methods" refer to these built-in features. There may also be "attributes" specific to each class that serve to differentiate them.

****

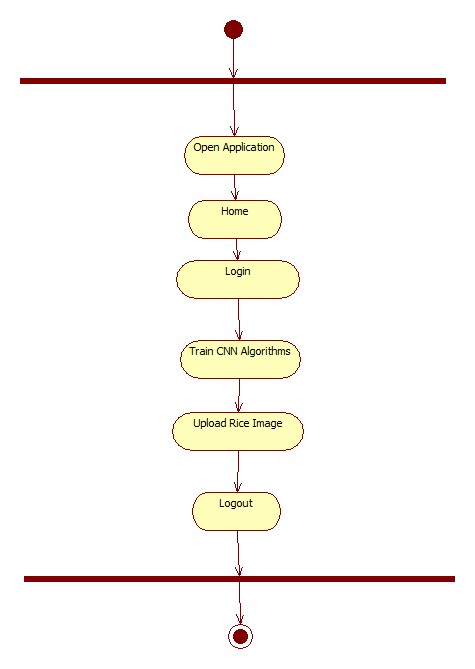
**Object diagram:**

Class diagrams have a variant known as the object diagram. A class is a type of object. An object is said to represent a class's status at a specific instant in the operating system. The object diagram depicts the current state of the system's various classes and their connections at a specific point in time.

****

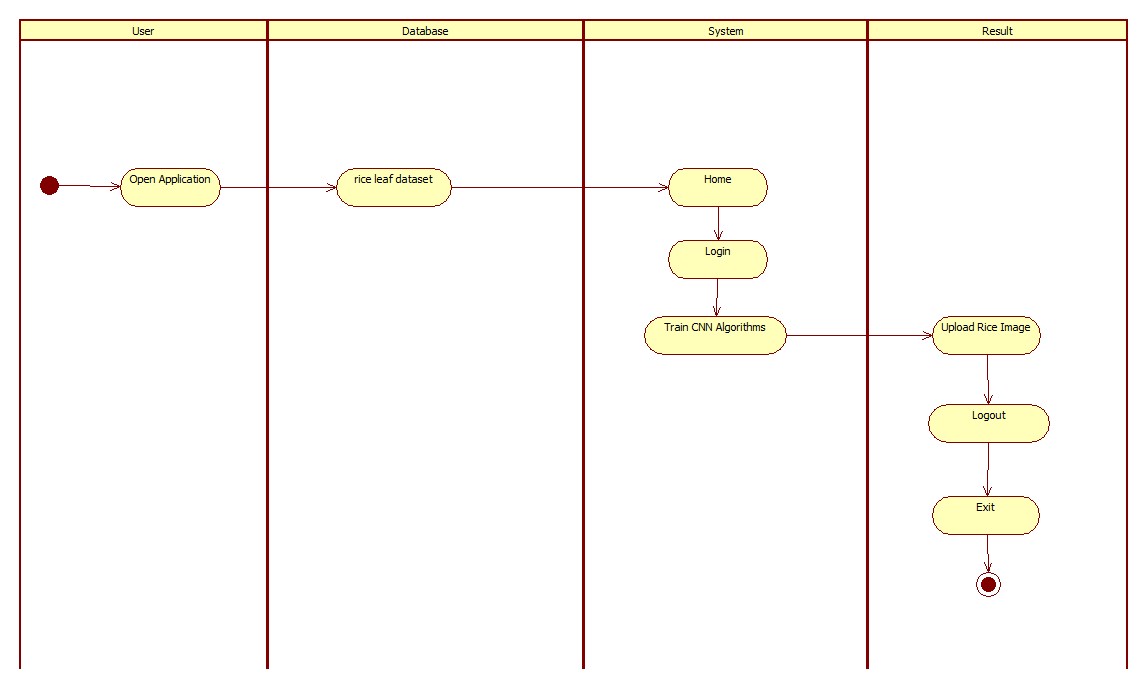
**State diagram:**

When you draw a state diagram, you're depicting all of the numerous states that various system objects can be in at a given time. When an event occurs, objects in the system alter their states. State diagrams are also used to show how an object's current state changes over time, in response to events in the system.

****

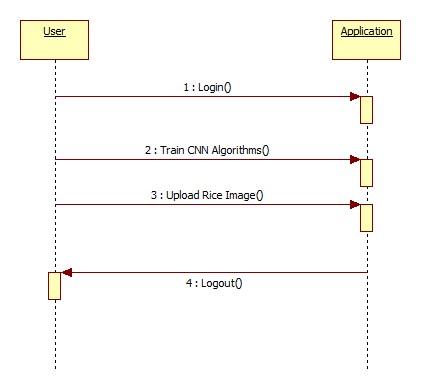
**Activity diagram:**

Using an activity diagram, you can see how the system works. There are similarities between an activity diagram and a state diagram in that they both comprise activities and actions as well as beginning and ending states as well as guard conditions.

****

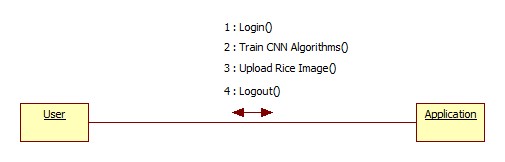
**Sequence diagram:**

A sequence diagram is a visual representation of the system's interactions. A sequence diagram's most crucial feature is that it's time-ordered. In other words, the interactions between the items are represented step by step in this model. Messages are the means by which various elements of the sequence diagram communicate with one another.

****

**Collaboration diagram:**

These relationships are depicted in a collaboration diagram. A numbering system aids in the tracking of the interactions' order in the database. Using the cooperation diagram, it is possible to see all the ways in which one object can interact with the other.

****

**4.3 IMPLEMENTATION:**

**MODULES:**

1. **Experimental Setup**

A 64-bit Windows 10 PC was used in the experiment. The CNN model was built with Keras 2.2.4 deep learning framework, TensorFlow 1.14.0 backend, and Python 3.7.2.

B. Processing of images.

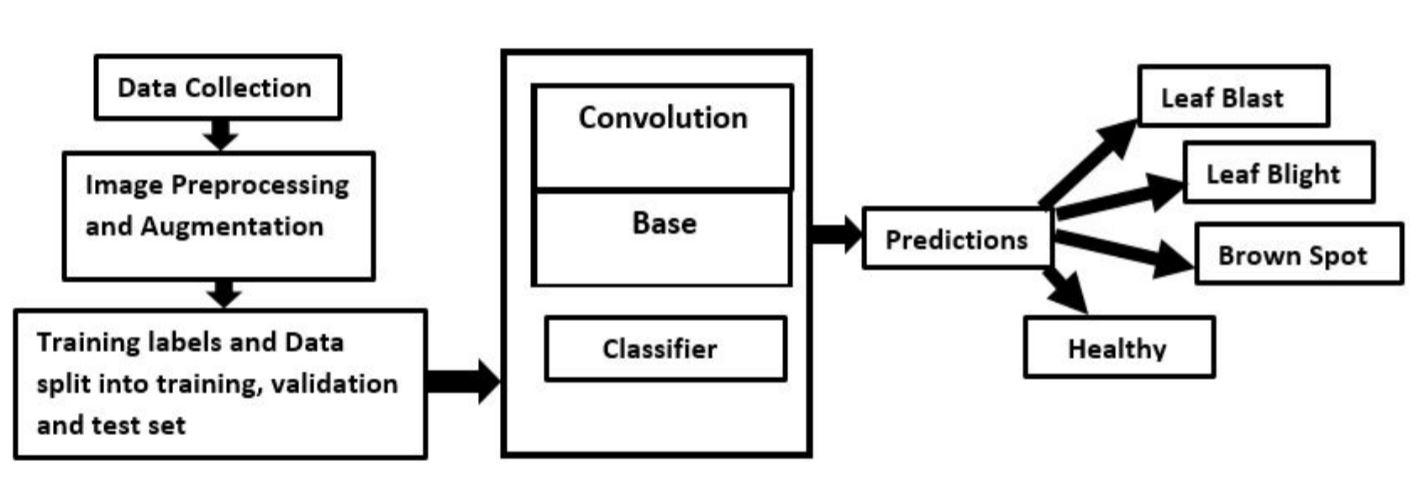
In addition to being taken in the field, all of the images were taken online. The dataset description includes images of leaf blast, leaf blight, brown spot, and healthy plants. Image Enhancement and Retouching As a result of applying various enhancement techniques such as zoom, rotation, and horizontal and vertical shift to the collected photos with ImageDataGenerator in Keras, new images are generated at 224\*224 pixel resolution.

C. CNN's Modeling School

The picture data set must be loaded in order to do training and testing. Class labels and images are kept in distinct arrays for training reasons. Due to the train-test split technique, training consumes 70% of the data, while testing consumes 30%. 30% of the data is utilised for validation and 70% of it is divided up again after that. The class labels are encoded as integers using a one-hot encoding procedure, and each label is represented as a vector rather than an integer. Finally, the final fully linked layers are erased from Keras. Additions to the system that cannot be trained are made. To finish, we applied a softmax filter to the feature extractor's flattened output before applying the softmax filter. Our model was designed from the ground up using the Adam optimizer with categorical crossentropy as the loss function for classification. Since the results remained stable after 25 epochs, we've stopped here. Figure 3 depicts the classification procedure steps that we've taken.

In support of the selected model, provide justification

What we call "transfer learning" is the process through which what we learn in one setting can be applied to a different one. Due to the fact that most real-world situations don't contain millions of labelled data points, transfer learning is extremely beneficial in training neural network models. In order to train a neural network from scratch, a large amount of data is required, however this data is not always available. Because the model has already been pre-trained, it is possible to build a robust machine learning model using only a small amount of training data. A VGG Net that has been pre-trained on our little dataset was utilised instead.

****

**Fig. 3. Overview of the steps of the proposed model**

**Algorithm:**

**convolutional neural network**

A deep learning technique known as Convolution Neural Networks is described in this study as an idea for automating government functions using Artificial Intelligence technology (CNN). People can read news and notifications about new government programmes on the Internet, and they can subsequently express their opinions about the programmes. This information can help the government make better judgments. Software like human brains are needed to identify public opinion about schemes automatically, and this software must be able to discern whether the opinions expressed are in favour of positive or negative.

The author's suggestion is to construct a CNN model that works like a human brain in order to build such an automated opinion detecting system. In order to make this CNN model perform like automated decision making without any human interactions, we can generate it for any service and implement it. According to author's concept of multiple models in which one model can identify or recognise human handwritten digits and the other model can detect sentiment from text sentences that can be offered by humans about government initiatives, this technique is already being proposed. We've included a new model in our extension that can identify emotion in a facial image as part of our extension model. Using facial expressions rather than words or sentences is a better way to convey feelings. So our new study can anticipate the emotions of people based on their faces.

As a way to explain how a convolutional neural network based image classifier works, we will develop a six-layer neural network that can detect and differentiate one image from the other. In order to run this network on a CPU, we'll need to design a very small network. In order to train a traditional neural network on a standard CPU, it would require a lot more parameters and take a long time. However, our goal is to demonstrate how to use TENSORFLOW to construct a real-world convolutional neural network.

Mathematical models for solving optimization problems are what neural networks are really all about. It is the basic unit of computation in neural networks, which is why they are formed of neurons. If an input (say, x) is fed into an active neuron, the neuron does some computations on it (like as multiplying it by a variable called w and then adding another one called b), the result of which is a new value (such as z=wx+b). Non-linear activation function f is used to produce the ultimate output of a neuron, which is called activation. Activation functions come in a variety of shapes and sizes. Sigmoid is a well-known activation function. The sigmoid neuron is the neuron that uses the sigmoid function as an activation function. Neurons are given names based on their activation roles, and there are numerous varieties to choose from, like RELU and TanH.

This is the next building element of neural networks, and it is called a layer when neurons are stacked in one line. See the layered image below for more information.



**5.SOFTWARE ENVIRONMENT**

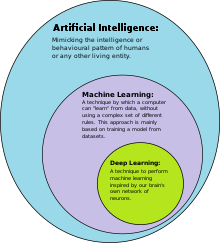
**Deep Neural Networks**

Deep learning (sometimes known as "deep structured learning") is a machine learning method that is part of a broader family. It's possible to learn under close supervision, in a semi-unsupervised setting, or on one's own.

From computer vision and speech recognition to natural language processing and machine translation to bioinformatics, medication creation and medical image analysis, many different types of deep learning architectures, such as convolutional neural networks and recurrent neural networks, have been employed.

We built artificial neural networks (ANNs) based on the dispersed communication and information processing found in biological systems. There are numerous ways in which artificial neural networks (ANNs) differ from biological brains. Artificial neural networks are static and symbolic in comparison to the dynamic and analogue (plastic) brains of most organic animals.

In deep learning, there are several levels in the network. Nonpolynomial activation functions with one hidden layer of unbounded breadth can be universal classifiers, according to early research. An infinite number of bounded-size layers, on the other hand, is a recent version that allows for practical application and optimal implementation while maintaining theoretical universality under moderate conditions. These layers of deep learning can be heterogeneous and deviate from biologically-informed connectionist models because of this "organised" element

****

The goal of deep learning is to learn structured representations of large datasets layer by layer, which is accomplished by creating multiple layers of neurons in the brain.

The network's architecture:

Models based on networks

Models of intelligence called "deep neural networks" were developed mathematically to resemble the functioning of the human brain.

• A network model is a way of describing how various layers of a network interact with one another.

• The following are some of the questions to consider when developing network models:

Other questions include: How many neurons should be used in each layer, as well as how the layers should be arranged?

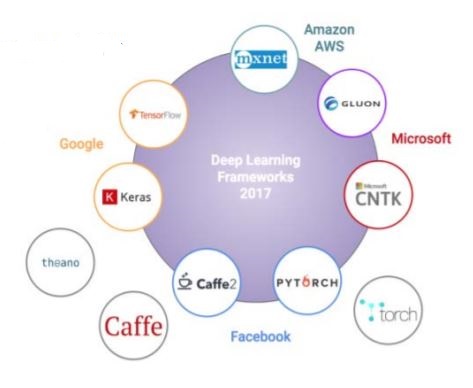
There are a number of typical CNN models that function well for a wide range of common issues. AlexNet, GoogleNet, Inception-ResNet, VGG, and so on are examples.

The frameworks for deep learning.

Because of its complexity, developing a deep learning solution is a major undertaking.

• A framework is a set of tools that makes it easier to create comprehensive learning solutions.

Using frameworks makes complex programming jobs easier.



**Popular Frameworks:**

Based on Github ratings and forks as well as Stack Overflow activity, TensorFlow is the most widely used deep learning framework.

For CNN modelling (imaging/computer vision applications), as well as its Model Zoo, Caffe was developed by the Berkeley Vision and Learning Center (BVLC) (a selection of pre-trained networks) All these frameworks are accompanied by interfaces that are wrapped around one or more frameworks. ' Keras is the most well-known and commonly used deep learning interface today. An API for deep learning built in Python, Keras is a high-level API.

Stream of deep learning

A framework for development must be chosen before any work can begin.

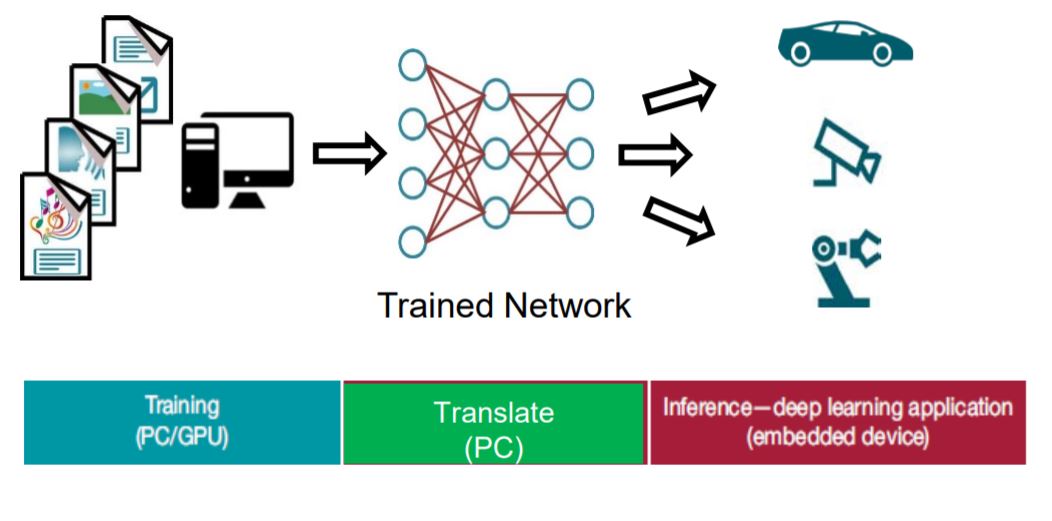
choosing classes to train the network on from a labelled data set 2.

Making the first draught of a network diagram

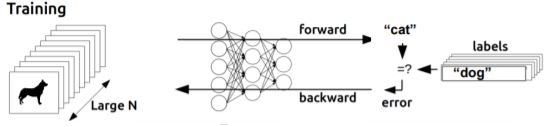
In addition, it is important to train the network.

5. Creating a binary file with the parameters and architecture.

a conclusion can be drawn



Training data:



**Introduction to deep learning summary**

• Exactly what is "deep learning"? Artificial intelligence (AI) is a catch-all term for any computer software that performs intelligently (AI). Artificial Intelligence (AI) is a subset of machine learning, which is a subset of deep learning as well.

• Deep learning is based on neural networks.

• Pattern recognition is the goal of neural networks, which are algorithms loosely built after the human brain.

• Deep learning: network types, nodes, layers, frameworks, and network models.

• The process of developing deep learning solutions

• Application areas

CNN:

Sparse interactions, parameter sharing, and analogous representations are all key concepts in convolutional learning. As a further benefit, convolution allows for working with inputs of any size. Matrix multiplication by a matrix of parameters with a different parameter for each input unit and each output unit is used in traditional neural network layers. To put it another way, every output unit has a direct connection to every input unit.

Reduce the number of parameters in the model to save memory while also increasing the model's statistical performance. As a result, fewer processes are required to calculate the final result. Efficiency gains of this magnitude are not uncommon. To get good results in many real applications, it is possible to store several orders of magnitude less data. These simple building blocks can then be used to construct complex interactions between numerous variables, allowing the network to explain these relationships in a more efficient manner than would otherwise be possible.

Deep Learning's Future

Scaling up deep learning systems often results in significant gains in performance. They generally do much better with a lot more data and a lot more computation. When compared to the number of synapses in the human brain, the GPT-318 language model has 175 billion parameters, but it produces considerably better text than the GPT-2 model, which has just 1.5 billion parameters. Meena and BlenderBot, two chatbots, keep getting better as they grow in popularity. Massive work is now being put into scaling up, which will greatly enhance present systems, but as has been discussed below, scaling cannot address the underlying flaws in current deep learning.

Several improvements to artificial intelligence can be identified by comparing human learning abilities to those of present AI.

1. Model-free reinforcement learning needs a lot more trial and error in comparison to supervised training. It seems that humans are capable of generalising fairly well, despite their lack of experience.

Whereas human beings need only a few examples to quickly adapt to changes in distribution, present systems are less resilient.

Current deep learning techniques excel at problems involving perception and what is known as system 1. Deep learning is a viable solution for system 2 operations that require a lot of planning. Where the problem lies. It has been the subject of machine learning theory since its conception because the id assumption argues that test examples are expected to come from the same distribution as training examples. Many agents' behaviours and a learning agent's ever-expanding horizon mean that assuming the universe is static in reality isn't feasible. The performance of AI systems degrades as soon as they leave the lab and enter the real world.

A subset of the more general goal of reducing sample complexity is to achieve greater robustness (also known as "out-of-distribution generalisation") when confronted with a new task, like transfer learning or lifelong learning, or even just a shift in distribution or the relationship between world states and rewards (the number of examples needed to generalise well). For model-free reinforcement learning, each rewarded trial provides less information about the task than each tagged example, which is required by existing supervised learning systems. Other than the usual id-generalisation, many studies show that humans can correctly interpret novel combinations of existing concepts, even if those combinations are extremely unlikely according to our training distribution, provided they respect high-level syntactic and semantic patterns we've already learned. Comparing neural network topologies has helped to clarify this ability to generalise.

**Introduction to Python:**

ABC was a precursor to Python, and Python is the successor to Python. In the year 1991, Guido Van Rossum came up with the idea. Code in this language is easy to understand thanks to the large amount of whitespace. It is an interpretive, general-purpose, and high-level programming language. In order to enable a variety of programming paradigms, an object-oriented approach was used. Python is a dynamically typed and garbage-collected programming language that supports both procedural and functional programming. This was followed by the release of Python 2.0 in the year 2000.

python 3.0 was released in late 2008.

Python's formatting was clean and the English keywords were used more frequently. The visual structure of a programme typically depicts the program's semantic structure. The whitespace indentation is used instead of curly brackets and semicolons to delimit blocks in this language. The indentation parameters for the blocks are as follows:

Indentation will rise when certain statements are made

This indicates the end of the current block by reducing the indentation.

1. Statements:

There is no need to declare the data type when assigning a value to a variable. The allocation of storage to names and objects can be caused by assigning a common value to several variables in succession..

• The if statement, along with the else block and the elif (i.e., else if) block, will surely be executed.

In general, "import" statements are used to import modules whose variables and functions can be utilised in the current application. There are three ways to specify this:

i. Import module> as [formalname>].

from a module's \*import\* function

Invoke functions 1 and 2 from a module, respectively, as shown in Listing iii. 2Expressions:

The arithmetic operations are identical to those in other programming languages, with the exception of division, which is different. There are two basic sorts of divisions in Python:

i. The floor division (//) is

Division by a floating point number

The NumPy libraries utilise an infix operator that is denoted by @ for matrix multiplication starting with python 3.5.

Concatenation of tuples and stringformat are handled by + and percent, respectively, in Python.

Note:

\*\* is the operator used for exponentiation in this example.

It is possible to use () as a key for dictionaries because tuples are immutable.

It is impossible to use a list as a dictionary key because lists are mutable and can be denoted by the symbol [].

Each element of the tuple, list, or array is referred to as a "shallow copy" when it is returned by Slice(:). The start index will be used for the slice, but the stop index will be omitted.

3. Libraries:

The following ten libraries are the most frequently cited by machine learning programmers for inclusion in their work:

• TensorFlow

• Pandas

• Scikit-Learn

• NumPy

• Keras

• Pytorch

• LightGBM

• Eli5

• SciPy

• Theano

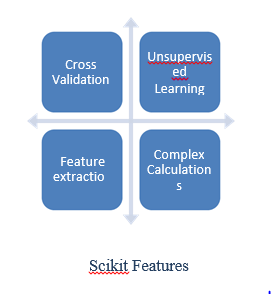
1. NumPy:

"Numerical Python" encompasses linear algebra, Fourier transforms, and other fundamental concepts. An array containing real numbers can be used to represent images and binary row streams in this library's Array interface. The following attributes are included:

### 

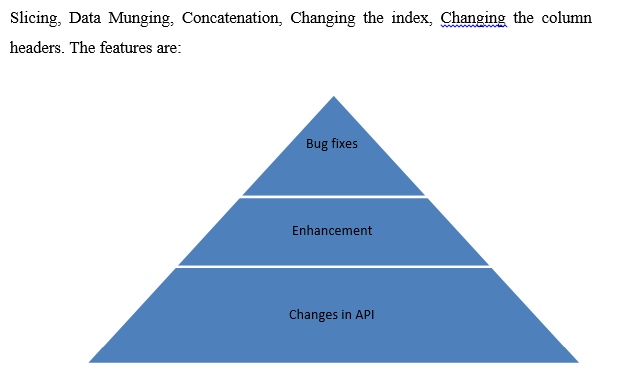
1. Scikit-Learn:

In general, a python library for dealing with complicated data is linked to Numpy and SciPy. This software is used mostly for classification, regression, clustering, and dimensionality reduction. To import modules from it, the name "sklearn" is given. The following attributes are included:



1. **Pandas:**

For the most part, this library focuses on providing high-level data structures and a number of tools for evaluating them. It is responsible for the processing and modification of structured data. Executed operations are described here.



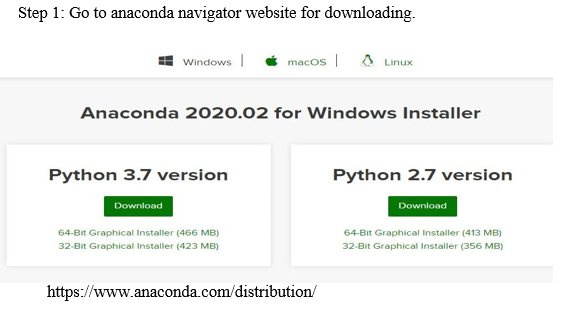
1. **Matplotlib:**

Python programmers can make use of this module to create 2D graphics. Useful in python, shell, web servers and other GUI toolkits as well. There are two significant drawbacks to this module: it relies heavily on other packages, and it only works with Python. In terms of plots, there are three categories:

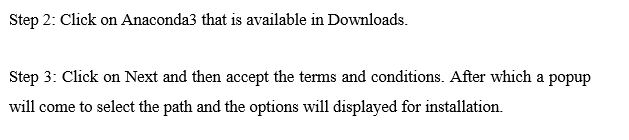
### 

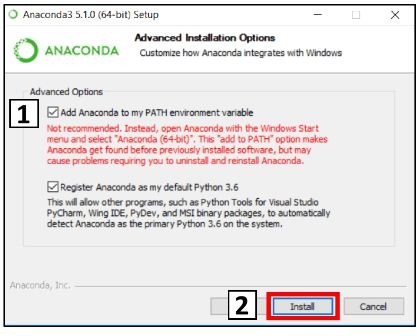
**ANACONDA( A PYTHONDISTRIBUTION):**

**Installation of Anaconda:**

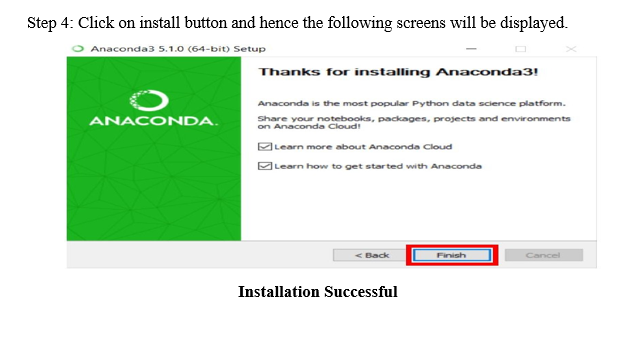


**Anaconda navigator installer**





### Installation options



**Introduction:**

For scientific computations, this is a free and open source distribution for R and Python computer languages aimed at streamlining the handling of packages and their deployment. The "conda" system, a package management system, is in charge of keeping track of the different versions. For MacOS, Linux, and Windows-based systems, this download comprises 1500 data science packages that were picked from the PyPI virtual environment manager and conda. You can use pip (recommended installation programme) or one of the other two types of managers.

When pip is used to install a package, any dependant python packages are also installed without the user's knowledge.

ii. conda:

This manager will thoroughly examine the existing environment, which includes all previously installed and currently running packages, before adding the one that's now missing. Installing conda is as simple as typing conda install.

You can use "pip" to install any accessible PyPI packages into the conda environment, and the conda will keep track of what is being installed. Python 3.7 will be installed by default.

Anaconda navigator is a desktop GUI (graphical user interface) that can launch and maintain applications and environments without the usage of command line commands.

### 

**6.SYSTEM TEST**

In order to discover errors, testing is important. In order to uncover all conceivable flaws or vulnerabilities in a product, testing is the process of looking for them. Checking the functionality of parts, subassemblies, assemblies, and/or a finished product can be done using this approach Exercises are performed to ensure that the software system satisfies its specifications and user expectations, and does not fail in an unacceptable manner. There are many different sorts of tests. Each form of test is designed to suit a certain testing need.

WHAT TESTS ARE AVAILABLE?

Unit tests

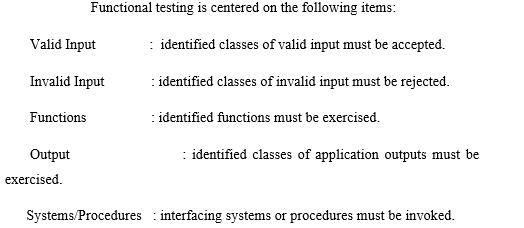
To guarantee that a program's fundamental logic is working effectively and that programme inputs create legitimate outputs, unit testing is utilized. Validate all decision paths and internal code flow. It is the testing of the application's individual software units. It is done prior to the integration of a single unit. This is an invasive structural test that demands prior understanding of the system's architecture. This sort of test is intended to evaluate a specific business process, application, and/or system configuration at the component level. There are a number of approaches to ensure that a business process adheres to its written criteria, including unit tests.

Testing of the system as a whole

The purpose of an integration test is to verify if two or more pieces of software can actually work together as a single unit. An event-driven method to testing focuses more on the basic outputs of screens and fields. Integration tests indicate that, despite the fact that the components were satisfied individually, the combination of components is proper and consistent. The goal of integration testing is to discover any difficulties that may occur as a result of merging distinct components.

a test of usefulness

Systems documentation, business and technical requirements, and user guides all play a role in functional testing since they demonstrate that the functionalities being tested are in fact present and accounted for.



Based on the requirements, critical functionalities, or specific test cases that need to be tested, functional tests are organised and created in accordance with these needs. The system's capacity to cover all important data fields, preset processes, and subsequent operations must also be verified. Before functional testing is finished, new tests are discovered and the value of the ones that already exist is assessed.

Verification of Input/Output

An integrated software system must meet a set of requirements before it can be tested to ensure it does. By putting a setup to the test, it may verify that it will provide the expected results. It is an example of systems testing to do a configuration-oriented system integration test. System testing is based on flow diagrams and explanations of how processes work, and the focus is on pre-driven process links and integration points.

Performing experiments in a black box

Testers who are familiar with the programme or its intended use conduct this type of testing. It's got a good point. This is the ideal tool for testing sites that can't be tested from the black box.

Testing in the Absence of a Test Environment

Software is tested using Black Box Testing, which means testers aren't privy to the module's internal structure, programming language, or workings. A definitive source document, such as a specification or requirements document, must be used to write black box tests, as with most other types of testing. Testing in which the programme being tested is considered as a black box is called "black box testing." You can't look inside it. The test gives inputs and responds to outputs without taking into account the software's functionality.

Testing of the system as a whole

As part of the software development lifecycle, unit testing is often performed in conjunction with coding and unit testing. Alternatively, unit testing can be performed as a separate process.

Techniques for doing testing Detailed functional tests are being prepared as well as manual field testing.

Goals of the exam

• Each and every field entry must function correctly.

• Pages can only be accessed by clicking on the link.

This includes the entry screen, messages, and responses.

Aspects that will be examined

Inspect the data to make sure it is in the correct format.

• There should be no room for duplicate entries.

It's important that all links point to the relevant page.

Integrity Verification

Integrating software components one at a time to look for bugs in the interfaces that could lead to errors is known as software integration testing.

Testing the interoperability of software components or applications, such as those in a software system or at the enterprise level, is the goal of an integration test.

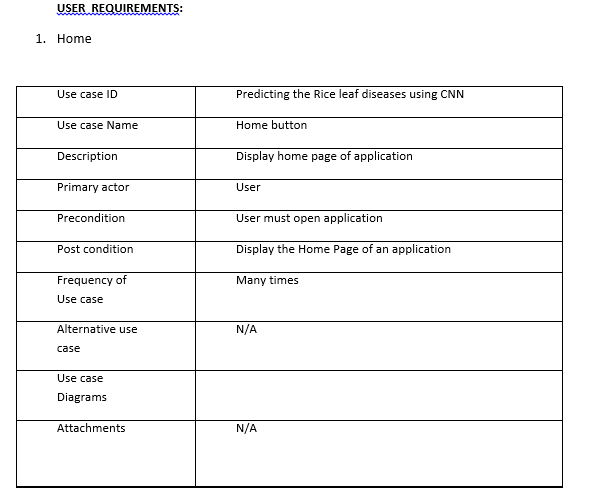
All of the test cases listed above passed with flying colours. So far, there haven't been any issues.

The Process of Performing Acceptance Tests

Acceptance of the user Any project's testing phase necessitates substantial input from the project's end users. It also ensures that the system's functional requirements are met, as well.

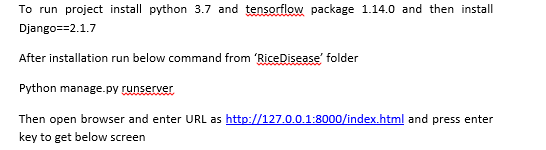
Results: Each of the above-mentioned test cases was a complete success. There were no issues.

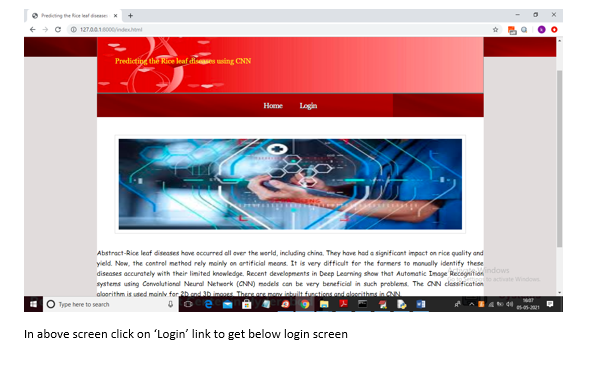
**Test Cases**

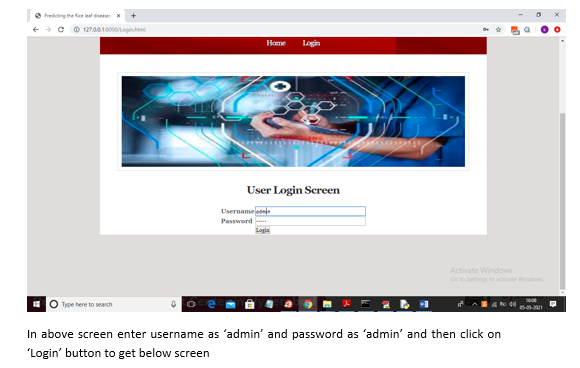


|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test | Test Case | Test Case | Test Steps | | | Test | Test |
| Cas | Name | Description | Step | Expected | Actual | Case | Priorit |
| e Id |  |  |  |  |  | Statu | Y |
|  |  |  |  |  |  | s |  |
| 01 | Start the | Host the | If it | We | The | High | High |
|  | Applicatio | application | doesn't | cannot | application |  |  |
|  | N | and test if it | start | run the | hosts |  |  |
|  |  | starts |  | applicati | success. |  |  |
|  |  | making sure |  | on. |  |  |  |
|  |  | the required |  |  |  |  |  |
|  |  | software is |  |  |  |  |  |
|  |  | available |  |  |  |  |  |
| 02 | Home Page | Check the | If it | We | The | High | High |
|  |  | deployment | doesn’t | cannot | application |  |  |
|  |  | environmen | load. | access | is running |  |  |
|  |  | t for |  | the | successfully |  |  |
|  |  | properly |  | applicati | . |  |  |
|  |  | loading the |  | on. |  |  |  |
|  |  | application. |  |  |  |  |  |
| 03 | User | Verify the | If it | We | The | High | High |
|  | Mode | working of | doesn’t | cannot | application |  |  |
|  |  | the | respond | use the | displays the |  |  |
|  |  | application |  | Freestyle | Freestyle |  |  |
|  |  | in freestyle |  | mode. | Page |  |  |
|  |  | mode |  |  |  |  |  |
| 04 | Data Input | Verify if the | If it fails | We | The | High | High |
|  |  | application | to take the | cannot | application |  |  |
|  |  | takes input | input or | proceed | updates the |  |  |
|  |  | and updates | store in | further | input to the |  |  |
|  |  | the | the |  | database. |  |  |
|  |  | database. | Database |  |  |  |  |

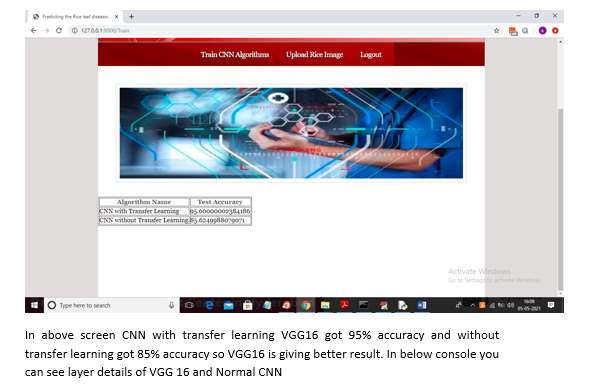
**7.SCREENSHOTS**

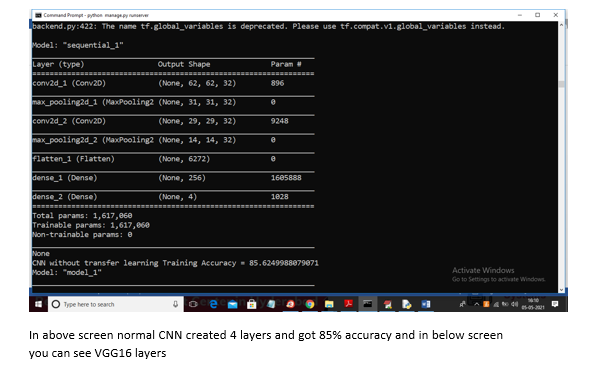


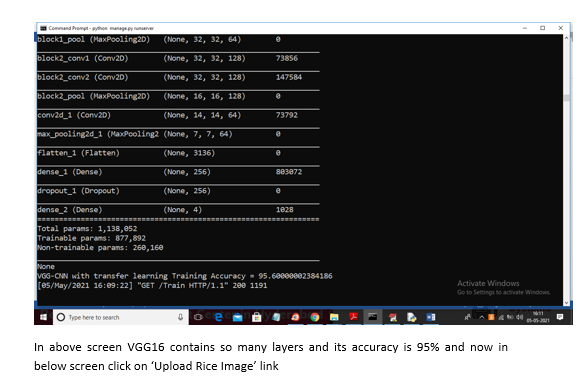


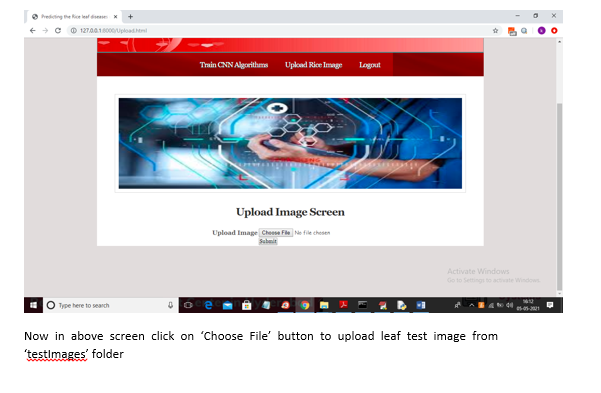


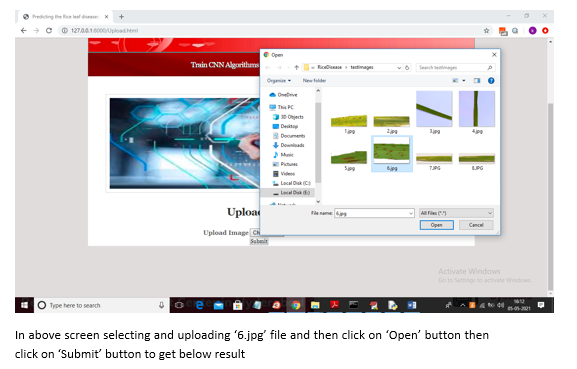


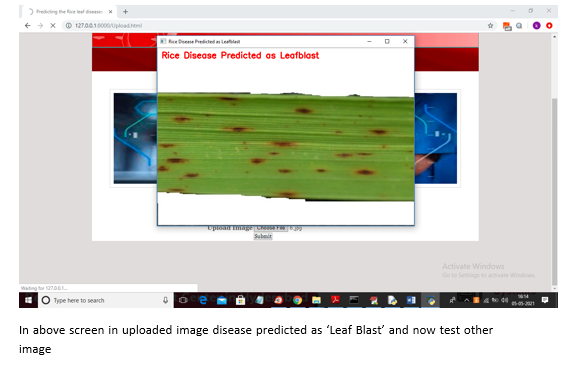


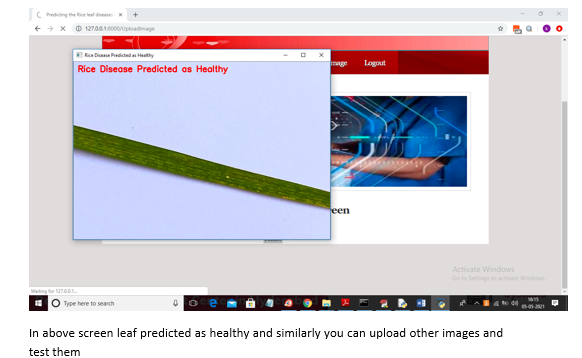








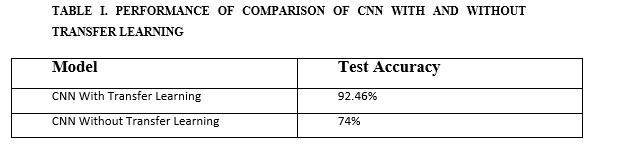


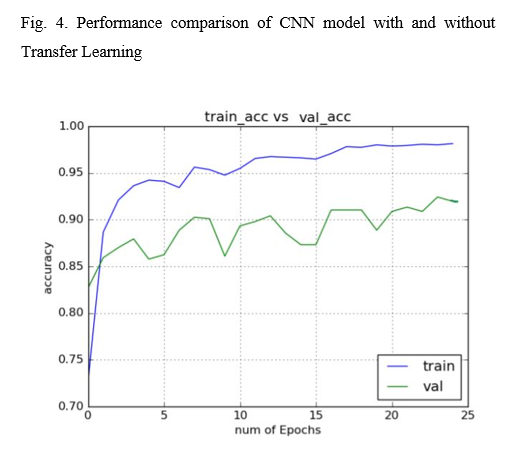


**RESULT**

* 1. **Calculations**

The suggested model properly predicted 97% of the training data and 95% of the test data after running for 20 epochs on 150 training data. A CNN model without transfer learning was trained and tested using the same data set, but with different split ratios. Optimization settings including batch size, epochs and rmsprop were fine-tuned, however the maximum accuracy was still only 74% with batch sizes of 16, 30, and 0.4 epochs respectively. In the CNN model without transfer learning, ReLU, Maxpooling, and dropout layers are followed by two Fully Connected Layers and SoftMax. Figure 4 in Table I compares suggested CNN models with and without Transfer Learning. Figure 5 displays the CNN's training and validation accuracy, as well as the number of epochs, using Transfer Learning.

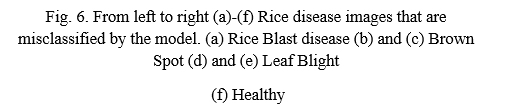






B. Analysis of Errors Figure 6 shows images that are incorrectly categorised using the proposed CNN model (a)-(f). The misclassifications for each disease type are detailed in the following section. Although it belongs to Rice Blast, image (a) is assigned to Brown Spot due to the image's blurriness and pixelation. Small brown patches on the same rice leaf may be the cause. Despite being labelled "Healthy," images (d) and (e) in this set depict leaf blight. Poor lighting and blurred images could be to blame. Healthy: The image (f) is in good health, however it has been labelled as Brown Spot due to its poor contrast and blurriness. Images (b) and (c) belong to Brown Spot, but are categorised as Blast by the EPA. Small blast lesions may be a possible cause of this. Blast lesions seem as brown spot lesions in (d).

****



**8.CONCLUSION**

In this research, we present a deep learning architecture that classifies 95% of the test photos correctly after training on 40 photographs of rice leaves and then testing on another 20 images. As a result of fine-tuning the VGG16 model, we were able to significantly increase the model's performance on such a short dataset. We capped the number of epochs employed at 20 after receiving data showing no improvement in accuracy or decrease in loss on either the training or validation sets.

In the future:

More photos from agricultural areas and agricultural research organisations are needed in the future to further increase the accuracy of our results. In the future, we want to include a cross-validation mechanism to further verify our findings. It would also be beneficial if we could compare the findings achieved with those of more advanced deep learning models and other current efforts in progress. Other plant leaf diseases, which are significant crops in India, can be detected using the model described here.

**9. REFERENCES**

1. T. Gupta, ”Plant leaf disease analysis using image processing technique with modified SVM-CS classifier,” Int. J. Eng. Manag. Technol, no. 5, pp. 11-17, 2017.

2. Y. Es-saady,T. El Massi,M. El Yassa,D. Mammass, and A. Benazoun, ”Automatic recognition of plant leaves diseases based on serial combination of two SVM classifiers,” International Conference on Electrical and Information Technologies (ICEIT) pp. 561-566, 2016.

3.P. B. Padol and A. A. Yadav, “SVM classifier based grape leaf disease detection,” Conference on Advances in Signal Processing (CASP), pp. 175-179, 2016.

4. L. Liu and G. Zhou, “Extraction of the rice leaf disease image based on BP neural network,” International Conference on Computational Intelligence and Software Engineering ,pp. 1-3,2009.

5.S. Arivazhagan and S.V. Ligi, “Mango Leaf Diseases Identification Using Convolutional Neural Network,” International Journal of Pure and Applied Mathematics, vol. 120,no. 6, pp. 11067-11079,2008.

6.B. Liu,Y. Zhang, D. He and Y. Li, “Identification of Apple Leaf Diseases Based on Deep Convolutional Neural Networks Symmetry,” X. X. &Suen, C. Y. A novel hybrid CNN–SVM classifier for recognizing handwritten digits. Pattern Recognition,vol. 45,pp. 1318–1325,2012.

7.Y. Lu, S. Yi,N. Zeng,Y. Liu, and Y. Zhang, “Identification of Rice Diseases Using Deep Convolutional Neural Networks,” Neurocomputing, 267, pp. 378-384,2017.

8.R. R. Atole, D. Park, “A Multiclass Deep Convolutional Neural Network Classifier for Detection of Common Rice Plant Anomalies,” International Journal Of Advanced Computer Science And Applications,vol. 9,no. 1,pp. 67–70,2018.

9.V. Singh, A. Misra, “Detection of Plant Leaf Diseases Using Image Segmentation and Soft Computing Techniques,” Information Processing in Agriculture,vol. 4 ,no. 1,pp. 41–49,2017.

10. P. Konstantinos Ferentinos, “Deep Learning Models for Plant Disease Detection and Diagnosis,” Computers and Electronics in Agriculture ,vol. 145,pp. 311–318,2018.

11. S. P. Mohanty, D. P. Hughes, M. Salath´e, “Using Deep Learning for Image-Based Plant Disease Detection,” Frontiers in plant science ,vol. 7.

12.S. Xie, T. Yang, X. Wang, and Y. Lin, “Hyper-class augmented and regularized deep learning for fine-grained image classification,” IEEE Conference on Computer Vision and Pattern Recognition,CVPR2015,pp. 2645–2654,June 2015.

13. Y. Lu, S. Yi, N. Zeng, Y. Liu, Y. Zhang, ” Identification of rice diseases using deep convolutional neural networks”, Neurocomputing 267,pp. 378–384,2017.

14. K. Simonyan, A. Zisserman, “Very Deep Convolutional Networks for LargeScale Image Recognition”, Arxiv preprint arXiv: arXiv,pp. 1409- 1556,2015.