CROP DISEASE PREDICTION USING IOT AND DEEP NEURAL NETWORKS

A PROJECT REPORT

Submitted by

Anushka S [Reg No: RA1911042010019]

Anushka Agarwal [RA1911042010032]

Under the Guidance of

Dr. Kanmani P
(Assistant Professor, Department of Data Science and Business Systems)

In partial fulfillment of the Requirements
for the Degree of

BACHELOR OF TECHNOLOGY
COMPUTER SCIENCE AND BUSINESS SYSTEMS

DEPARTMENT OF DATA SCIENCE AND BUSINESS SYSTEMS

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603203

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR – 603 203

BONAFIDE CERTIFICATE

Certified that this B.Tech project report titled "**CROP DISEASE PREDICTION USING IOT AND DEEP NEURAL NETWORKS**" is the bonafide work of Ms. Anushka S [Reg. No.: RA1911042010019] and Ms. Anushka Agarwal [Reg. No.:RA1911042010032] who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on thebasis of which a degree or award was conferred on an earlier occasion for this or any other candidate.

DR. P KANMANI
SUPERVISOR
Assistant Professor
Department of Data Science and Business Systems

DR. M. LAKSHMI
HEAD OF THE DEPARTMENT
Department of Data Science and Business Systems

SIGNATURE OF INTERNAL EXAMINER

SIGNATURE OF EXTERNAL EXAMINER

Department of Data Science and Business Systems

SRM Institute of Science and TechnologyOwn

Work Declaration Form

Degree/ Course : B.Tech in Computer Science and Engineering with a specialization in Business Systems

Student Names : Anushka S, Anushka Agarwal

Registration Number: RA1911042010019, RA1911042010032

Title of Work : Crop Disease Prediction Using IOT And Deep Neural Networks

We hereby certify that this assessment compiles with the University's Rules and Regulations relating to Academic misconduct and plagiarism, as listed in the University Website, Regulations, and the Education Committee guidelines.

We confirm that all the work contained in this assessment is our own except where indicated, and that we have met the following conditions:

- Clearly references / listed all sources as appropriate
- Referenced and put in inverted commas all quoted text (from books, web, etc.)
- Given the sources of all pictures, data etc. that are not my own

- Not made any use of the report(s) or essay(s) of any other student(s) eitherpast or present
- Acknowledged in appropriate places any help that I have received fromothers (e.g. fellow students, technicians, statisticians, external sources)
- Compiled with any other plagiarism criteria specified in the Coursehandbook / University website

I understand that any false claim for this work will be penalized in accordance with theUniversity policies and regulations.

DECLARATION:
We are aware of and understand the University's policy on Academic misconduct and plagiarism and We certify that this assessment is our own work, except where indicated by referring, and that We have followed the good academic practices noted above.
If you are working in a group, please write your registration numbers and sign with the date for every student in your group. ANUSHKA S [RA1911042010019] ANUSHKA AGARWAL [RA1911042010032]

ABSTRACT

Crop disease is a major concern for farmers as it can lead to significant crop yield and quality losses. Traditional methods of examination through eyes is the primary method for illness detection, however, it can be laborious and labor-intensive. To overcome these challenges, researchers have developed a new approach that uses IoT technology and deep neural networks to predict crop diseases. Agricultural pests and illnesses significantly diminish yields, often by more than 50% or even causing crop failure. Precision farming now has new dimensions thanks to farming made possible by technology assisted by IoT and image-processing tools for predicting illness. This approach provides a more accurate and efficient way of detecting crop diseases and allows farmers to take preventative measures to protect their crops from disease.

The suggested approach integrates IoT and using image processing robust learning models to do classification or supports enhanced production by assisting in the prediction of agricultural disease. In conclusion, integrating IoT and deep neural networks in crop disease prediction can potentially revolutionize the agricultural industry. By providing accurate and timely predictions of crop diseases, farmers can take preventative measures to protect their crops and increase their yield and quality. Since the deep learning models used in this method were trained on huge datasets of images of both healthy and diseased crops, they are capable of spotting even the tiniest indications of disease in plants. Farmers may safeguard their crops from illness and maintain good harvests by anticipating crop diseases early on and taking preventative steps, such as spraying fungicides or modifying irrigation levels.

This method can result in greater farming efficiency in addition to minimizing crop losses. Farmers can use resources like water, fertilizer, and pesticides more efficiently by receiving precise and early predictions of crop diseases. Farmers' businesses might become more profitable and sustainable as a result of cost reductions. In summary, combining IoT and deep neural networks for crop disease prediction possesses prospective to completely transform industrial market.

ACKNOWLEDGEMENT

We express our humble gratitude to Dr. C. Muthamizhchelvan, Vice-Chancellor, SRM Institute of Science and Technology, for the facilities extended for the project work and his continued support. We extend our sincere thanks to Dean-CET, SRM Institute of Science and Technology, Dr. T.V.Gopal, for his invaluable support. We wish to thank Dr. Revathi Venkataraman, Professor & Chairperson, School of Computing, SRM Institute of Science and Technology, for her support throughout the project work. We are incredibly grateful to our Head of the Department, Dr. M. Lakshmi, Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for her suggestions and encouragement at all the stages of the project work. We want to convey our thanks to our Project Coordinator, Dr. P. Rajasekar, Assistant Professor for their inputs during the project reviews and support. We register our immeasurable thanks to our Faculty Advisor, Dr. R. Rajkumar, Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for leading and helping us to complete our course. Our inexpressible respect and thanks to our guide, Dr. Kanmnai P, Assistant Professor, Department of Data Science and Business Systems, SRM Institute of Science and Technology, for providing us with an opportunity to pursue our project under his mentorship. She provided us with the freedom and support to explore the research topics of our interest.

Anushka S [RA1911042010019]

Anushka Agarwal [RA1911042010032]

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ANN	Artificial Neural Networks
CNN	Convolutional Neural Network
IOT	Internet Of Things
RNN	Recurrent Neural Network
ANN	Artificial Neural Network
ROC	Receiver Operating Characteristic
GANs	Generative adversarial networks
AUC	Area Under the Curve
AI	Artificial Intelligence

CHAPTER 1

INTRODUCTION

1.1 General

Farming was economically unavailable and ineffective due to traditional farming practices, a decline in farm labor availability, a water shortage problem, and deteriorating soil conditions. Also, it is the ideal time to concentrate on preserving the environment and practice farming with modern technology to feed the current and future populations. The population will double in 2050, the United Nations Food and Agricultural Organization predicts. Agriculture's increasing output will give the country a consequential economic boost. Effective and timely farm operations are essential for agriculture, and incorporating technology innovations can provide suitable methods that enable multiple cropping, leading to increased yield and convenience.

Consumers should only receive nutrients from farm products, ensuring a healthy lifestyle. It is evident that crop diseases have gotten out of hand to the point that they are reducing the quantity and quality of agricultural crops. So, it is necessary for professionals to identify these issues, which might be costly. In other locations, farmers must travel great distances in order to contact professionals and are unable to contact them. For farmers, the traditional methods of identifying signs of illness in crops can be both costly and time-consuming. However, with the application of IoT and machine learning, it is now possible to determine crop health and detect diseases through methods such as regression and other machine learning algorithms. Crops need to be watched carefully since even one damaged crop can cause illnesses to spread to other crops.

The agricultural industry is experiencing a growing interest in technology and automation, which can be utilized to address various stages of the production cycle. With numerous variables affecting each decision, the product a farm's life cycle can be complicated process. The employment of various sensors, cameras, and IT units to monitor crops allows for the regular capturing of images, cutting-edge machine learning, and artificial intelligence techniques are being utilized to incorporate data gathered from various sources into imaging systems, with the aim of improving yield and reducing crop failure.

It is expected that in the future, the integration of the Internet of Things and image processing into agricultural machinery will further enhance the efficiency and accuracy of crop disease detection and prediction and may eliminate weeding and crop monitoring tasks that currently require manual labor. By incorporating cutting-edge technology into traditional farming methods, we can produce high-quality, high-yield goods.

Precision plant protection techniques are becoming more widespread, and a market for various technologically enhanced farming methods is emerging in the field of precision agriculture. Crop protection is achieved by integrating image processing methods like a study of colors and threshold values to identify or categorize plant illness. Artificial neural networks are presently employed as one of many possible methods for detecting plant diseases. ANN is a piece of information- the processing paradigm employed in the fields of cognitive science and machine learning is inspired by the processing of information mechanisms in biological nerve systems, including those in the brain The brain is made up of numerous, intricately connected neurons which cooperate to find solutions to particular issues.

The employment of various sensors, cameras, and IT units to monitor crops allows for the regular capturing of images, cutting-edge machine learning, and artificial intelligence techniques are being utilized to incorporate data gathered from various sources into imaging systems, with the aim of improving yield and reducing crop failure. It is expected that in the future, the integration of the Internet of Things and image processing into agricultural machinery will further enhance the efficiency and accuracy of crop disease detection and prediction and may eliminate weeding and crop monitoring tasks that currently require manual labor.

1.2 OBJECTIVE

The objective of crop disease prediction utilizing IOT and deep neural networks is providing a more precise and effective method of identifying agricultural illnesses, allowing farmers to take preventative actions to safeguard their crops and improving production both quantity their farm. Traditional identification of illnesses techniques, such as visual inspection, can be labor- and time-intensive and frequently not particularly accurate. Data may be gathered and analyzed in real-time by combining IoT technology and deep neural networks, giving farmers important knowledge on the health of their crops.

The initial goal was creating an IoT-based sensor network which can be used to gather information on several crop health-related metrics, including temperature, humidity, and soil moisture levels. Deep neural networks may then be used to analyze this data and find patterns and trends in crop diseases. Deep neural networks can precisely detect and forecast agricultural illnesses using image processing and machine learning methods, enabling farmers to quickly take preventive action to save their crops.

Objectives of Crop Disease Prediction Using IOT And Deep Neural Networks are:-

- Create an IoT-based sensor network to gather data

Creating a sensor network that can gather information on many aspects of crop health, such as temperature, humidity, and soil moisture levels, is the initial goal. This information may be used to track the condition of crops and spot any disease symptoms. The sensor network consists of a network of sensors that are positioned across the farm at various areas, and these sensors gather data on a regular basis. Using IoT technology, the data gathered by the sensors is then sent to a central hub where it is stored and examined. The sensor network enables real-time crop health monitoring, which is essential for spotting any early symptoms of illness. With the aid of this real-time monitoring, farmers can act quickly to stop the spread of illness and save their crops. In order to assist farmers create more successful disease management plans, the data gathered by the sensor network may also be utilized to spot trends and patterns in crop diseases. Overall, creating an IoT-based sensor network is an important step in the process of predicting crop diseases. Real-time data collection is made possible by it, giving farmers important information on the health of their crops. Farmers may take proactive steps to safeguard their crops, stop the spread of disease, and increase yields and quality by using this data to make educated decisions.

- Analyze data using deep neural networks

Utilizing deep neural networks to analyze the information gathered from the sensor network is the second goal. Deep neural networks can precisely detect and forecast agricultural illnesses using image processing and machine learning methods, enabling farmers to quickly take preventive action to save their crops. The data collected by the sensor network is typically in the form of images or time-series data, such as temperature and humidity readings. These data types are fed into the deep neural network, which processes the data using various layers of artificial neurons to learn the patterns and relationships between different variables. Once the deep neural network has been trained on a dataset of images or time-series data, it can then be used to predict the presence of crop diseases in new data. The user interface is designed to be intuitive and user-friendly, with a simple and easy-to-navigate layout. It should be accessible from a range of devices, including smartphones, tablets, and desktop computers. The interface may also include visualization tools, such as charts and graphs, to help farmers understand the data and make informed decisions. Developing a user-friendly interface is critical to the success of crop disease prediction using IoT and deep neural networks. By providing a platform that is easy to use and understand, farmers can quickly and efficiently access important information about the health of their crops. This can help farmers take timely action to prevent the spread of disease and protect their crops from damage. Additionally, a user-friendly interface can help to increase adoption rates of the technology among farmers, ultimately leading to increased crop yields and improved crop quality.

- Develop a user-friendly interface

The third goal is to provide a user-friendly interface that farmers may utilise to get real-time information on their crops. Farmers should be able to get extensive information on the condition of their crops using this interface, including any symptoms of illness. It should also offer advice to farmers on how to take precautions to safeguard their crops, such modifying watering schedules or using fungicides. Overall, analyzing data using deep neural networks is a critical objective of crop disease prediction using IoT and deep neural networks. By leveraging the power of artificial intelligence, we can accurately predict crop diseases and take timely preventative measures to protect crops from damage. The use of deep neural networks in crop disease prediction offers several advantages over traditional methods. For example, deep neural networks can process large amounts of data quickly and accurately, allowing for the detection of crop diseases in real-time. Additionally, deep neural networks can identify patterns and relationships that may not be apparent to the human eye, enabling more accurate disease detection.

- Improve the efficiency and sustainability of farming operations

The ultimate goal is to give farmers fast and accurate information in order to increase the sustainability and efficiency of farming operations. We can increase the effectiveness and sustainability of farming operations while lowering costs and environmental impact by enabling farmers to make educated decisions. An essential goal of crop disease prediction utilizing IoT and deep neural networks is to increase the productivity and sustainability of agricultural operations. The productivity and sustainability of farming operations can be significantly impacted by crop diseases, which can result in yield and quality losses, increased pesticide and fertilizer usage, and decreased profitability. Farmers can preserve their crops, increase the efficiency and sustainability of their operations, and accurately anticipate crop illnesses using IoT and deep neural networks.

Reducing the time and resources required for disease identification and control is one-way crop disease prediction may increase efficiency. Visual examination is the primary technique used in conventional disease detection techniques, which can be labor- and time-intensive. Deep neural networks and IoT-based sensor networks can help farmers identify agricultural illnesses more quickly and accurately, allowing them to take prompt preventive action to save their harvests.

- Facilitate decision-making for farmers

  Making judgments on how to manage their crops in the midst of uncertainty is one of the major difficulties that farmers face. It can be challenging for farmers to determine when and how to take action since crop diseases can be unexpected and spread swiftly. To give farmers the knowledge they need to make knowledgeable decisions about their crops, deep neural networks and IoT are being used to facilitate decision-making for farmers. The system must give farmers access to real-time information on the condition of their crops and the presence of any illnesses in order to accomplish this goal. This information needs to be highly sensitive and particular, as well as accurate and dependable. Additionally, the information must be presented in a simple, understandable manner, with visualizations and alerts to draw attention to any potential problems. Farmers may use this information to make well-informed decisions about how to manage their crops once they have access to it. To stop the spread of disease, they could elect to use fungicides or other treatments, or they might decide to take other steps like altering irrigation methods or planting dates.

1.3 SCOPE

The scope of "crop disease prediction using IOT and deep neural networks" is to accurately forecast crop diseases by utilizing deep neural networks that have been trained using data from IoT devices and sensors to monitor crop health and environmental variables. Early disease diagnosis, enhanced crop output, less pesticide usage, and greater agricultural efficiency are all benefits of this strategy. It has great potential to advance sustainable farming methods and has a favorable influence on the agriculture sector. In the old days, farmers would personally check their crops for disease symptoms and then take appropriate action depending on their findings. It requires an extensive amount of work and attention to detail to use this strategy to identify illnesses at an early stage when they are most curable.

Real-time data collection on crop health and environmental variables is possible with IoT devices, sensors, and deep neural networks. Temperature, humidity, soil moisture, and other elements that may have an impact on crop development are among the data that sensors may collect. To identify possible illnesses and forecast their spread, this data may be put into deep neural networks that have been trained on massive databases of crop disease trends. This strategy has a lot of advantages. Farmers who can identify infections early can act rapidly to stop the spread of the illness and minimize crop losses.

This strategy may also result in more environmentally friendly agricultural methods by lowering the usage of pesticides and other chemicals, which may be expensive for farmers and have negative environmental effects. By making educated choices about irrigation, fertilization, and harvesting, farmers may optimize crop productivity and decrease waste with the aid of accurate disease prediction.

The following are some of the major spheres where this technology can make a difference:

- Early disease detection

Using IoT and deep neural networks to forecast agricultural illnesses has several advantages, one of which is the capacity to identify diseases at an early stage. This can assist farmers in acting swiftly to stop the disease's spread and reduce crop losses. Deep neural networks are used to analyse data after sensors continually monitor crops in order to find patterns and anomalies that could point to the existence of illness. A major advantage of crop disease prediction utilising IoT and deep neural networks is early disease identification. Detecting diseases promptly enables landowners to act swift steps to halt illness's stretch or cut back on farming losses. This is crucial since it can be challenging and expensive to control a disease after it has established itself. IoT-based sensors may be installed in the field to continually monitor crop health while gathering information on the weather, soil moisture, and other important factors. Deep neural networks are then used to analyze this data in order to find patterns and anomalies that could point to their existence of illness.

Deep neural networks have benefit of processing enormous volumes of data and spotting tiny patterns that human observers would miss. This implies that even little changes in crop health can be recognized, acting as a possible disease early warning system.

- Precision agriculture

Precision agriculture, which uses data-driven methods to optimize crop output, is a bigger trend that includes crop disease prediction utilizing IoT and deep neural networks. This technology can assist to increase agricultural yields and quality while lowering resource waste by giving farmers precise and timely information on the health of their crops. Precision agriculture is a method of farming that maximizes crop output, minimizes waste, and lessens the impact of farming on the environment. Because it gives farmers precise and timely information about crop health, crop disease prediction using IoT and deep neural networks is a crucial part of precise farming. Information about weather, moisture, soil wetness, and other environmental elements that might have an impact on crop development and health may be collected using IoT-based sensors. Deep neural networks are used to analyze this data and find patterns and abnormalities that might be signs of illness. Farmers may maximize crop yield and minimize waste by utilizing this data to make educated decisions regarding irrigation, fertilization, and other inputs. Using IoT and deep neural networks to anticipate crop diseases can also assist farmers in minimizing the use of pesticides and other chemicals, which can be detrimental to the environment and public health. Farmers may decrease the need for pesticides and other chemicals, resulting in more sustainable and ecologically friendly agricultural practices, by identifying illnesses early and taking the right measures. As a result, farming operations become more effective and productive by enabling farmers to make better decisions about planting, harvesting, and other tasks.

- Sustainable agriculture

Crop disease prediction utilizing IoT and deep neural networks can help create a more sustainable agricultural system by increasing the effectiveness of farming operations and lowering the usage of chemical inputs. In light of climate change and mounting demands on natural resources, this is especially crucial. Another crucial use of IoT and deep neural networks for crop disease prediction is sustainable agriculture. Sustainable agriculture is the practice of growing crops while reducing their harmful effects on the environment and enhancing social and economic well-being. IoT and deep neural network-based crop disease prediction can support sustainable agriculture in a number of ways. Farmers can minimize crop losses and cut back on the use of pesticides and other chemicals via early and precise disease detection. This can lessen the detrimental effects on the health of agricultural workers and the local population while preserving the natural ecology and biodiversity of the surrounding area. Crop disease prediction utilizing IoT and deep neural networks can aid farmers in minimizing the usage of pesticides while simultaneously maximizing the utilization of water, fertilizer, and other inputs. Farmers may identify fields that need more or less water, fertilizer, or other inputs by using data collected by IoT sensors and analyzed by deep neural networks, minimizing waste and promoting more effective resource use. In order to be sustainable, agriculture must also support social and economic well-being. Crop disease prediction utilizing IoT and deep neural networks can help farmers boost their yields and revenues, improving economic viability by minimizing crop losses and lowering waste.

- Improve the efficiency and sustainability of farming operations

The ultimate goal is to give farmers fast and accurate information in order to increase the sustainability and efficiency of farming operations. We can increase the effectiveness and sustainability of farming operations while lowering costs and environmental impact by enabling farmers to make educated decisions. One of the main goals of crop disease prediction utilizing IoT and deep neural networks is in order to boost performance and sustainability of agricultural operations. Utilizing these technologies, farmers can choose their crops with more knowledge, use resources more effectively, and produce less waste, leading to more productive and environmentally friendly agricultural methods. Reducing the need for human labor is one-way crop disease prediction may boost farming operations' efficiency. Traditional disease detection methods rely on visual inspection by skilled professionals, which may be labor- and time-intensive. Farmers may lessen the need for physical inspection by employing IoT sensors and deep neural networks to identify illnesses early and correctly, and instead concentrate their efforts on adopting preventative actions to safeguard their crops. Predicting crop diseases can also assist farmers in making the most use of resources like water, fertilizer, and pesticides. Farmers may identify fields that need more or less water, fertilizer, or pesticides by analyzing data from IoT sensors and deep neural networks, minimizing waste and promoting more effective resource usage. This can not only raise the profitability and sustainability of farming operations but also cut expenditures.

1.4 INNOVATION

With usage of IoT and deep learning models for crop disease prediction, which offers an accurate and effective method of identifying crop illnesses, is where the novelty in the project "crop disease prediction using IoT and deep neural networks" resides. The system's development seeks to stop crop loss from disease and help farmers by offering a practical fix. The usage of Reference Architecture for the Internet of Things, which enables cloud-based and on-field disease detection algorithms to anticipate sickness, is one notable advance. Another innovation that improves prediction accuracy is the use of neural networks to train the system on illness categorization and prevention.

Additionally, standardizing the data and using pixel characteristics from farmer-taken photos to narrow the range of returned values is another creative strategy that improves the system's effectiveness. Another novelty is the classification of the neuronal network with the aid of a back-propagation neural network and giving of a diagnosis and treatment strategy to the farmers. Another creative strategy for improving the system's user-friendliness is to add forums for farmers to discuss information based on their locality and local language support and speech-to-text capability to the app.

The system is trained on illness categorization and prevention using neural networks. High accuracy rates of 93% or above are seen in the performance measures for each model developed for various plants, which is essential for machine learning algorithms. Early disease identification, which enables farmers to take precautions to safeguard their crops from disease, is one of the main advantages of this discovery. As a result, crops produce bigger yields and better quality products, enhancing the nation's food security. By requiring less time and labor than conventional techniques of disease diagnosis, the technology also has the potential to increase the productivity and sustainability of farming operations.

To sum up, the project "Crop Disease Prediction Using IoT and Deep Neural Networks" innovates by combining deep neural networks with IoT technologies to precisely and effectively identify crop illnesses. By enabling early disease diagnosis, enhancing the sustainability and efficiency of farming operations, and emphasizing accessibility and inclusivity for farmers, the system has the potential to completely transform the agricultural sector.

CHAPTER 2

LITERATURE SURVEY

Crop diseases are a serious danger to the agricultural sector because they result in large financial losses and lower crop yields. An expert eye examination is still used in traditional crop disease diagnosis procedures, but it's time-consuming, costly, and frequently wrong. Deep neural networks and IOT have created new opportunities for more precise and effective agricultural disease prediction. Deep neural networks and IOT have created new opportunities for more precise and effective agricultural price forecasting. Deep neural networks and the Internet of Things have created new opportunities for more precise and effective agricultural price forecasting. Real-time data from IoT devices, such as weather stations, soil moisture sensors, and satellite photos, is essential for predicting crop prices since they can vary quickly owing to a variety of variables, including climate change, weather patterns, and market trends. On the other hand, a deep neural networks are a class of machine learning methods is made to resemble the structure and operation of the human brain. They are excellent for forecasting agricultural prices because they can analyze big datasets and spot intricate trends. In order to estimate agricultural prices, IoT sensors and deep neural networks are used. This method offers a number of advantages, including better accuracy, increased efficiency, and a decreased reliance on expert judgment. Informed decisions regarding planting, harvesting, and selling crops may be made by farmers with the aid of accurate crop price forecasts, resulting in more productive and environmentally friendly agricultural methods. Crop price forecasting can also help decision-makers make educated choices about market laws, trade, and agricultural policies. But there are also difficulties that must be overcome, including the need for high-quality data, data privacy, and security issues, and the requirement for data format standardization. To fully utilize the potential of this technology, more study is required. Additionally, the system makes use of farmer-taken photos' pixel characteristics, which are later standardized to narrow the range of returned values. A backpropagation network is categorized using a neural network of neurons, and farmer is then given a diagnosis and treatment strategy as a consequence.

2.1 IoT in Crop Disease Prediction

In order to anticipate agricultural diseases, IoT devices gather information from a variety of sources, including weather stations, soil sensors, and crop photographs. Deep neural networks may be trained with this data to reliably forecast crop disease. Real-time data from IoT devices is essential for anticipating crop diseases since they can spread quickly owing to number of variables, such as climate patterns, state of ground, and methods for managing vegetation. IoT devices can forecast agricultural diseases more accurately than conventional approaches, according to many studies. Malvika Ranjan et al. suggested a method to identify plant illnesses using the picture of sick leaves in their experiment, "Detection and Classification of Leaf Disease Using Artificial Neural Network." In order to discriminate between healthy samples and unhealthy plants, Artificial Neural Networks (ANN) must be trained by carefully selecting feature values. The accuracy of the ANN model is 80%. Early disease identification, which enables farmers to take precautions to safeguard their crops from disease, is one of the main advantages of this discovery. As a result, crops produce bigger yields and better quality products, enhancing the nation's food security. By requiring less time and labor than conventional techniques of disease diagnosis, the technology also has the potential to increase the productivity and sustainability of farming operations. According to the research "Detecting unhealthy leaf regions and classifying plant leaf diseases using texture features," According to S. Arivazhagan, the process of identifying a disease consists of the following four basic steps: The input RGB picture is first given a color transformation structure, and green pictures are then selectively removing with predetermined criterion rating. Volume 7, Issue 07, 2020 1607 of the European Journal of Molecular & Clinical Medicine (ISSN 2515-8260) discovered and uninvolved, then comes division procedure, or the appearance metrics are calculated to generate useful pieces. Therefore, a categorizer is employed to categorize illness using attributes that were extracted.

Compared to conventional approaches, using IoT sensors to anticipate agricultural diseases provides a number of benefits. IoT devices, for instance, may gather data in real time, giving a more precise view of the crop's actual status. IoT devices may also collect data from several sources and span wide regions, which can give a more thorough knowledge of the elements that lead to agricultural diseases. Finally, IoT devices can lessen the need for manual labor, improving the effectiveness and cost-effectiveness of crop disease prediction.

Overall, the use of Internet of Things (IoT) devices for agricultural disease prediction has considerable promise for revolutionizing the agriculture sector by enabling more precise and effective disease prediction, which would promote more environmentally friendly farming methods and increase crop yields. For instance, soil sensors may be used to track nutrients, pH, and moisture levels, which can be used to forecast the risk that a crop would contract a certain disease. Similar to this, weather stations may be used to track temperature, humidity, and other weather-related variables that might assist forecast the risk that a crop would contract a certain disease.

2.2 Deep Neural Networks in Crop Disease Prediction

A subgroup of machine learning techniques called deep neural networks is created to mimic the composition and operation of the brain of an individual. These are perfect for forecasting agricultural diseases because they can analyze huge information and spot complicated patterns. Using historical data on crop diseases, weather patterns, soil quality, and other pertinent aspects, deep neural networks may be taught. Numerous studies have demonstrated that deep neural networks can predict crop diseases more accurately and efficiently than conventional techniques. It has been demonstrated that deep neural networks are useful for predicting agricultural diseases. In order to effectively anticipate the incidence of crop diseases, these models can learn to recognize patterns and correlations in massive datasets of crop photographs, meteorological data, and other pertinent information.

Deep neural networks handle enormous volumes of data by processing progressively complicated characteristics of the incoming data through numerous layers of linked nodes. In order to produce precise predictions concerning crop diseases, these models may be trained to extract pertinent aspects from crop photos, such as leaf shape, texture, and color, and integrate them with additional data, such as weather conditions and soil properties. Technique for determining plant diseases using Generative Adversarial Networks in the article has been proposed "Plant disease detection using CNN and GAN" by Emaneul Corts. Background segmentation is carried out to ensure accurate feature extraction and output mapping. It seems clear that employing Gans might be useful for categorizing plant diseases, however, background-based segmentation had no discernible effect on precision. Jyotsna Bankar et al. suggested using the Inception v3 model to categorize mammals into distinct specie in their study titled "Convolutional Neural Network based Inception v3 Model for Animal Classification". Inception v3 has the capacity to both categorize objects, which makes it useful for many different image classifiers.

Deep neural networks have capability to learn from a variety of data sources, which enables them to capture complicated relationships between many elements that contribute to crop disease. A deep neural network, for instance, may be trained to anticipate the appearance of a certain crop disease using a mix of weather, soil, and crop imagery while taking into consideration the intricate interactions between these several components. The capacity of deep neural networks to process vast volumes of data is another benefit. This is crucial for agricultural disease prediction, because precise models must be trained using massive quantities of relevant data, including crop photographs. Deep neural networks can effectively analyse vast volumes of data, enabling them to spot subtle patterns and associations that conventional statistical approaches can overlook.

Deep neural networks has been successfully utilized previously several years to forecast agricultural diseases in a number of different ways. For instance, deep neural networks have been used by researchers to accurately forecast the incidence of wheat illnesses using hyperspectral photographs of wheat fields. Similar levels of accuracy have been attained by other researchers using deep neural networks to predict the existence of tomato illnesses from photos of tomato leaves. A technique to spot plant illness using Generative Adversarial Networks was suggested under article "Plant disease detection using CNN and GAN" by Eaneul Cor. It seems clear that employing Gans might be useful for categorising plant diseases, however background-based segmentation had no discernible effect on accuracy.

To sum up, the project "Crop Disease Prediction Using IoT and Deep Neural Networks" innovates by combining deep neural networks with IoT technologies to precisely and effectively identify crop illnesses. By enabling early disease diagnosis, enhancing the sustainability and efficiency of farming operations, and emphasizing accessibility and inclusivity for farmers, the system has the potential to completely transform the agricultural sector. Deep neural networks possess capacity to learn from a variety of data sources, which enables them to capture complicated relationships between many elements that contribute to crop disease. A deep neural network, for instance, may be trained to anticipate the appearance of a certain crop disease using a mix of weather, soil, and crop imagery while taking into consideration the intricate interactions between these several components. The capacity of deep neural networks to process vast volumes of data is another benefit.

2.3 Applications of Crop Disease Prediction using IoT and Deep Neural Networks

Crop disease prediction utilizing IoT and deep neural networks has a wide range of uses. Supporting farmers in selecting crop management strategies is one such use. A more sustainable approach to farming can be achieved by minimizing the use of pesticides and other chemicals with accurate crop disease prediction. Crop disease forecasting can also help decision-makers make educated choices on market laws, trade, and agricultural policies. Crop disease prediction utilising IoT and deep neural networks has a wide range of possible uses. One of this technology's main advantages is its early detection of agricultural illnesses, which enables farmers to take preventive action before the disease spreads. This may entail applying pesticides selectively, changing watering and fertilisation routines, or even getting rid of sick plants. Farmers can lessen the overall spread of infections and their effects on crop output by spotting them early.

Precision agriculture can be made possible through use of IoT and deep neural networks in crop illness prediction, in addition to early detection and prevention. Farmers may make data-driven decisions on crop management practises by gathering and analysing data from a range of sources, including crop photographs, weather conditions, and soil properties. This may entail adopting precision pesticide treatment, changing planting densities, and optimising irrigation and fertilisation schedules. Farmers may lower input costs while raising yields and overall crop quality by optimising these practises. The publication "Applying image processing technique to detect plant diseases" by Kulkarni et al. presents a method in the very beginning and precise detection of plant illnesses utilising ANN additional forms of image manipulation techniques. Recommended approach produces superior outcomes with up to 91% acceptance rate since it is based on both Gabor filter and an ANN classifier in obtaining features in categorization.

Promoting sustainable agricultural methods is a significant use of crop disease prediction utilising IoT and deep neural networks. Predicting crop diseases accurately can help farmers use fewer pesticides and other chemicals, resulting in more environmentally friendly agricultural methods. This may lessen the negative effects of agriculture on the environment while producing foods that are healthier and more nourishing. Farmers can also increase profitability while fostering sustainability by lowering input costs.

2.4 Challenges and Future Directions

While there are many advantages to using IoT and deep neural networks to predict crop diseases, there are some issues that must be resolved as well. High-quality data requirements, data privacy and security issues, and the requirement for standardising data formats are some of these difficulties. The creation of increasingly sophisticated deep neural networks, the incorporation of other technologies like blockchain, and the creation of user-friendly interfaces for farmers and other stakeholders are some of the future prospects for study in this field. The accessibility and calibre of the data represent one of the main obstacles. While there are many data sources, such as crop images, weather conditions, and soil characteristics, that can be used to predict crop diseases, their reliability and accessibility can vary. Access to some data sources can also be restricted due to data privacy issues, especially when it comes to farm-level data.

Despite these obstacles, the application of IoT and deep neural networks to the prediction of crop disease has a wide range of potential future directions. The creation of more reliable and accurate models that can take a greater variety of environmental elements into consideration is one possible option. Additionally, improvements in sensor technology might make it possible to collect data that is more precise and accurate, which would increase the accuracy of these models. The present technology has several drawbacks as well. For instance, deep learning models may need an abundance of information to train well, yet that data might not always be accessible. Furthermore, changes in weather patterns, soil characteristics, or other environmental factors may have an impact on the precision of these models, making it difficult to predict crop diseases under all conditions.

Additionally, the system makes use of farmer-taken photos' pixel characteristics, which are later standardised to narrow the range of returned values. A backpropagation network is categorized using a neural network of neurons, and the farmer is then given a diagnosis and treatment strategy as a consequence. For the purpose of predicting agricultural diseases, several case studies have been done to show how well IoT sensors and deep neural networks function. In one study, IoT devices and deep neural networks were used to forecast the development of fusarium head blight in wheat with a 97% accuracy rate. The creation of more reliable and accurate models that can take a greater variety of environmental elements into consideration is one possible option. Additionally, improvements in sensor technology might make it possible to collect data that is more precise and accurate, which would increase the accuracy of these models. A deep neural network was worked in a different prediction study the onset of rice blasts illness, with 95% accuracy.

Overall, even though using IoT and deep neural networks to predict crop diseases has its challenges and limitations, there are still significant potential advantages. This technology has the potential to revolutionise how farmers control crop diseases, resulting in more effective, sustainable, and lucrative agricultural practises. By tackling these issues and looking into new development opportunities.

2.5 Case Studies

For the purpose of predicting agricultural diseases, several case studies have been done to show how well IoT sensors and deep neural networks function. In one study, IoT devices and deep neural networks were used to forecast the development of fusarium head blight in wheat with a 97% accuracy rate. A deep neural network was deployed in a different prediction experiment the onset of rice blasts illness, with 95% accuracy. One noteworthy case study was carried out in China, where scientists created a deep learning model to forecast maize illnesses using pictures of the leaves. The model considerably outperformed conventional techniques of illness diagnosis, with an accuracy rate of 94%. In order to make it simpler for farmers to use the technology in their fields, the researchers also created an IoT system that could automatically gather and analyse photos of maize leaves.

Another case study was carried out in India, where scientists created an IoT system that could track soil moisture and crop health in real-time. Farmers could access data on crop growth, production, and disease risk thanks to the system's usage of sensors and data analytics. The technique has been proven to save water use and fertiliser application while increasing crop yields by up to 30%.The use of IoT and deep neural networks crop illness forecasting has also been investigated by researchers in United States. In one research, the development of soybean rust, a severe fungal disease that can drastically lower crop production, was predicted using a mix of drone footage, machine learning, and weather data.

Up to 14 days in advance, the researchers' precise predictions of the disease's spread gave farmers crucial time to take preventative action. Early disease identification, which enables farmers to take precautions to safeguard their crops from disease, is one of the main advantages of this discovery. As a result, crops produce bigger yields and better quality products, enhancing the nation's food security. By requiring less time and labour than conventional techniques of disease diagnosis, the technology also has the potential to increase the productivity and sustainability of farming operations.

These case studies show the potential of IoT and deep neural networks for predicting crop disease overall. One noteworthy case study was carried out in China, where scientists created a deep learning model to forecast maize illnesses using pictures of the leaves. These technologies have the potential to considerably increase crop yields, lower input costs, and encourage more sustainable agricultural practices by giving farmers real-time information on crop health, disease risk, and environmental conditions.

CHAPTER 3

PROPOSED METHODOLOGY

"Internet of Things" (IoT) system for crop identifying diseases is a multi-step method involving various techniques or technologies. One of the critical components of this system is the use of sensors and cameras the crop leaves in pictures. The collected images are employed to develop and evaluate deep-learning models that detect diseases in crops. The first stage of the process involves crop image acquisition using farm-installed cameras that take pictures of the crop leaves. The images are then pre-processed using a pre-processing model that performs noise reduction and data normalization. Feature normalization is also done in this step to convert the image vector into unit space. In the next stage, crop image analysis is performed to segment the photos and identify area of curiosity. The segmentation method used in case is area-based segmentation, which uses the color of the leaf to distinguish between the healthy and sick areas of the plant's leaf.

After accurately identifying the infected regions of the plant's leaf, the segmentation process proceeds, and the resulting segmented images are then utilized to train and test deep-learning models for crop disease detection. These models are designed to learn from the features of the images and identify patterns that are indicative of crop disease. Convolutional networks, recurrent networks, and support vector machines are among various techniques utilized by the models to label the pictures as "healthy" or “diseased”. IoT system for crop disease detection has many benefits for farmers, including accurate and timely detection of crop diseases, which can lead to increased crop yield and quality. The system also reduces the need for manual monitoring and visual inspection, which can be time-consuming and labor-intensive. Additionally, this approach can lead to more sustainable farming practices, as it reduces the need for pesticides and other chemicals, resulting in a more environmentally friendly approach to agriculture.

In conclusion, the integration of IoT and deep learning for crop disease detection has the potential to revolutionize the agricultural industry. By providing accurate and efficient detection of crop diseases, farmers can take preventative measures to protect their crops and increase their yield and quality. This approach can also lead to more sustainable and environmentally friendly farming practices, ultimately benefiting both farmers and the wider community.

Pre-processing will be applied to the picture pulled from the database and the camera. Following picture scaling, image enhancement, and the RGB picture are transformed after edge detection to a grayscale version since a grayscale image provides the highest degree of accuracy for defect identification. Various analytics approaches are applied to categorize photos based on specific issue in stake. Finally, images with identified diseases are uploaded employing optimization approaches, to the cloud. Real-time preceding labor already subsists, but not in the embedded context. This strategy's goal is to combine deep neural network modeling with contemporary image processing techniques. The procedure is simplified by the use of Internet of Things systems, which extends to some degree of automation for taking pictures on a regular basis. In turn, this enables the user to readily monitor environmental parameters without physically entering the field.

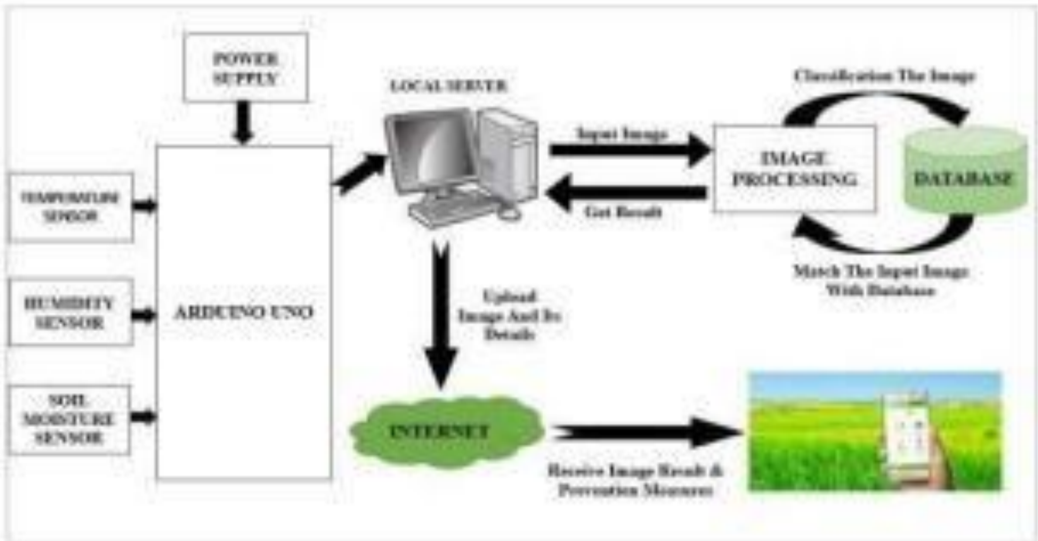


Fig 3.1 Proposed System Design

A neural network, which is composed of interconnected artificial neurons with weighted connections, is structured in layers based on predetermined patterns, with each neuron in a layer receiving input stimuli, analyzing it, and transmitting outputs to other neurons through connected links. To adapt to their contexts, networks mainly rely on learning. Neural networks are complex systems that consist of several components. The four primary components of a neural network are the input, node-to-node connectivity, activation function, and learning function. The input node receives signals and becomes active when triggered by incoming signals. The node-to-node connectivity determines how the nodes are connected and how the signals are propagated throughout the network.

The activation function is a rule that converts the input into an output inside each node. This output is then used as the input for the next node in the network. The learning function is responsible for controlling the weights of the input-output pairs, which allows the neural network to alter and enhance its progress in performance. The integration of the four components allows a neural network to acquire knowledge from.

3.1. DATASET

For this experiment, Sharada P. Mohanty et alPlantVillage .'s public dataset for plant leaf disease identification was employed [19]. The dataset comprises 87 000 RGB photos of healthy and diseased plants split into 38 groups of leaves. We have only chosen 25 classes to determine our method on. Table 1 displays these classes. The researchers decided to concentrate their investigation on 25 of these groups, which represented a variety of plant species and diseases kinds. In their study, Table 1 gave a comprehensive analysis of these 25 classifications, detailing the number of pictures in each class and the particular illnesses or disorders they represented.

This dataset allowed the researchers to train and test their deep learning model on a sizable and varied collection of photos, guaranteeing that it could successfully recognize a variety of plant illnesses. They used deep learning model called CNN, that is particularly effective at picture categorization tasks. Their research outlooked that their an improved deep learning model over traditional machine learning techniques at diagnosing plant illnesses with high levels of accuracy. Their research represents a significant advancement in the use of technology for evaluation and management of plant illness by utilizing strength for deep learning and the sizable dataset provided by PlantVillage.

Characteristics of dataset include:-

- Sum of 87,000 RGB pictures of both healthy and sick plants are included in the dataset.
- The photos are divided into 38 distinct leaf groupings that each depict a different plant species or type of illness.
- For their experiment, the researchers decided to concentrate on 25 of these groups, which were selected to symbolize range of plant species and disease types.
- Researchers can test their model on both sorts of images since the dataset contains both colour and grayscale photos.
- A number of sources, including both controlled situations and natural settings, were used to gather the photos in the collection.
- The dataset provides a variety of viewpoints for the model to learn from, including photographs of both close-up views of individual leaves and larger views of entire plants.
- The photos in the collection are of different quality and resolution, imitating the actual environmental factors that may be present for diagnosing plant diseases.

3.2 CROP DISEASE FORESEEABILITY

The capacity to anticipate the onset of crop illnesses before they actually develop is referred to as crop disease foreseeability. Predictive modelling, machine learning, data analysis are some of methods those may be used to achieve this. Farmers and other agricultural sector stakeholders can reduce the impact of crop illnesses on crop yields by taking preventive measures by being able to predict the emergence of crop diseases. Predictive modelling, machine learning, and data analysis are all techniques that may be employed to foretell the presence of crop diseases. These techniques entail the analysis of information on the environment, crop health, and other elements that might influence the emergence of crop diseases. Machine learning algorithms can forecast the risk of future disease outbreaks with a high degree of accuracy by finding patterns and connections in this data.

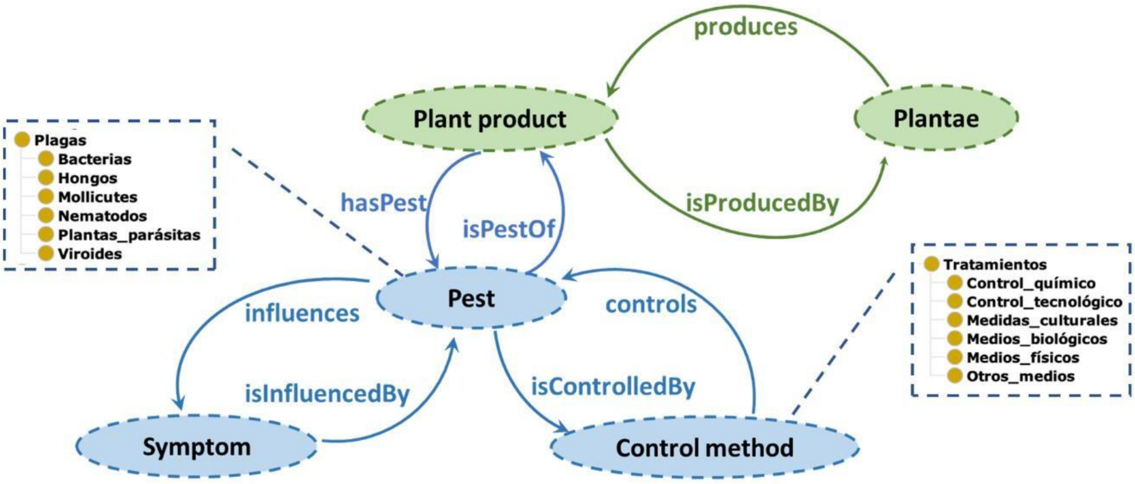


Fig 3.2. Crop disease control

In developing nations, where small-scale farmers may have little resources and be especially susceptible to crop losses, it is crucial to be able to predict crop diseases. Crop disease predictability can enhance food security and advance sustainable agriculture in these areas by helping farmers to anticipate and stop disease outbreaks.

3.3 DEEP NEURAL NETWORKS

Each layer of linked nodes or neurons in a DNN processes and refines the incoming data in successive levels. Each connection between a node in a layer and a node in a layer above or below has a weight that affects how strong a signal will be when it goes through that link. The weights of these connections are changed during training to improve the network's performance on a particular task, such as speech or picture categorization. The capacity of DNNs to automatically learn and extract features from the raw input data is one of its main advantages.

This indicates that they don't require human feature engineering and can be trained on enormous datasets. At number programmes, including computer vision, processing of natural language, voice detection, and predictive modelling, DNNs have shown to be quite successful. Farmers and other agricultural sector stakeholders can reduce the impact of crop illnesses on crop yields by taking preventive measures by being able to predict the emergence of crop diseases. Predictive modelling, machine learning, and data analysis are all techniques that may be employed to foretell the presence of crop diseases. These techniques entail the analysis of information on the environment, crop health, and other elements that might influence the emergence of crop diseases.

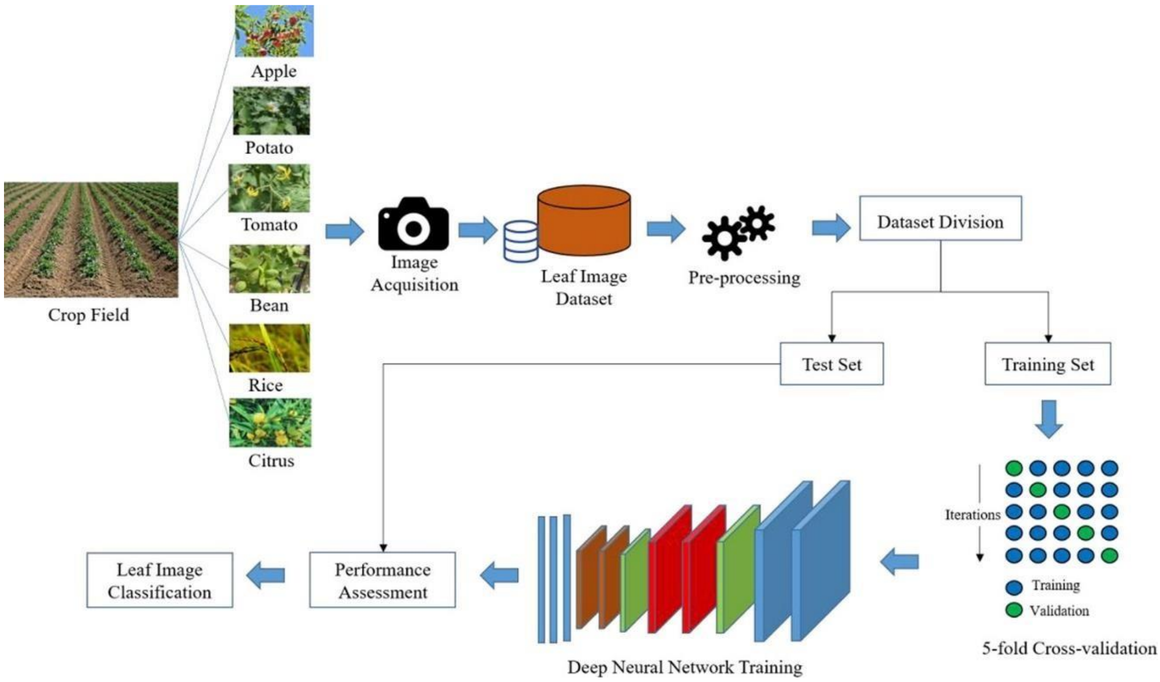


Fig 3.3. Dense convolutional neural networks

However, in order to achieve high levels of accuracy, DNNs need a lot of labelled training data, which can be computationally expensive. Furthermore, it can be challenging to comprehend how DNNs arrive at their predictions or decisions due to the complexity of their internal workings. Despite these difficulties, the impressive performance of DNNs on a variety of tasks has led to their rising popularity in recent years. They are being employed extensively in research and industry, and it's anticipated those they is going to continue performing significant part in the development of artificial intelligence.

3.4 RESNET-18

The ResNet series of deep neural networks, which includes ResNet-18, was developed to solve the issue of disappearing gradients in extremely deep networks. The introduction of residual connections, which enables the network to learn residual functions that may be added to the input of each layer, is the main innovation in ResNet18. This makes it easier for gradients to be back propagated across the network, which helps to solve the vanishing gradient problem. Additionally, because of the residual connections, deeper networks can be trained without experiencing performance degradation, which frequently happens with conventional CNNs.

ResNet-18's efficacy in predicting agricultural diseases has also been enhanced by combining it with other methodologies, including transfer learning and data augmentation. For instance, a work that was published in the Journal of Imaging improved ResNet-18 on a dataset of pictures of apple leaves with three distinct illnesses by using transfer learning. The researchers discovered that transfer learning considerably increased ResNet-18's classification accuracy, resulting in a test set accuracy of 98.2%.

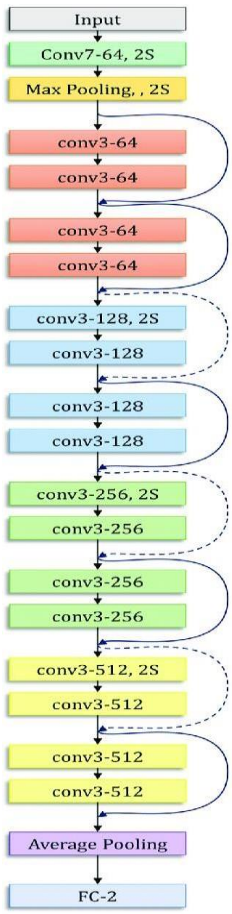


Figure 3.4. ResNet-18 Architecture Diagram

3.5. VGG-16

The 16-layer VGG-16 has 13 convolutional layers, 3 of which are completely linked levels, and other layers. Max pooling layers with a stride of 2 come after the convolutional layers, which utilize tiny 3x3 filters with a stride of 1. The network may learn more intricate elements from the input picture at each layer because of its architecture. The addition of user-friendly interfaces that can be customized to suit regional languages and speech-to-text capabilities is another noteworthy feature. This enables farmers to discuss information based on their locality in forums that may be added to the program. The significance of inclusion and accessibility in technological solutions for the agriculture sector is emphasized by this part of the project.

- Additionally, the system makes use of farmer-taken photos' pixel characteristics, which are later standardized to narrow the range of returned values.
- A backpropagation neural network is employed to categorize the web neurons, or farmer is then given a diagnosis and treatment strategy as a consequence.
- Early disease identification, which enables farmers to take precautions to safeguard their crops from disease, is one of the main advantages of this discovery.
- As a result, crops produce bigger yields and better quality products, enhancing the nation's food security.
- By requiring less time and labor than conventional techniques of disease diagnosis, the technology also has the potential to increase the productivity and sustainability of farming operations.

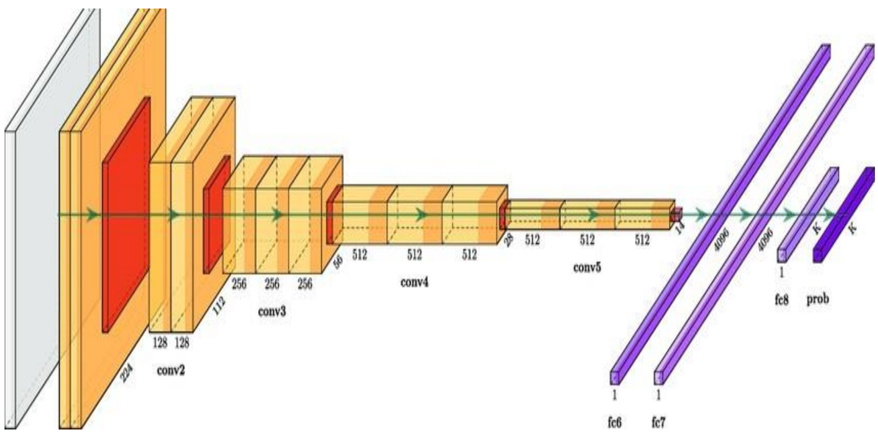


Figure 3.5. VGG-16 Architecture Diagram

3.6 CNN, DENSENET 121 and INCEPTION-V4

3.6.1 CNN

Christian Szegedy et al. presented the deep neural network architecture known as Inception-v4 in 2016. It is a member of the convolutional neural network family Inception, which is renowned for its efficiency in image categorization tasks. With the addition of various new features, Inception-v4 was created to enhance the functionality of its predecessors, Inception-v1, Inception-v2, and Inception-v3. The utilization of the Inception module, a building component made up of several parallel convolutional layers with various filter sizes, is one of the main aspects of Inception-v4. For tasks like picture identification and classification, this enables the network to collect characteristics of various sizes and complexity.

The addition of user-friendly interfaces that can be customized to suit regional languages and speech-to-text capabilities is another noteworthy feature. This enables farmers to discuss information based on their locality in forums that may be added to the program. The significance of inclusion and accessibility in technological solutions for the agriculture sector is emphasized by this part of the project. To sum up, the project "Crop Disease Prediction Using IoT and Deep Neural Networks" innovates by combining deep neural networks with IoT technologies to precisely and effectively identify crop illnesses.

By enabling early disease diagnosis, enhancing the sustainability and efficiency of farming operations, and emphasizing accessibility and inclusivity for farmers, the system has the potential to completely transform the agricultural sector. This enables farmers to discuss information based on their locality in forums that may be added to the program. The significance of inclusion and accessibility in technological solutions for the agriculture sector is emphasized by this part of the project.

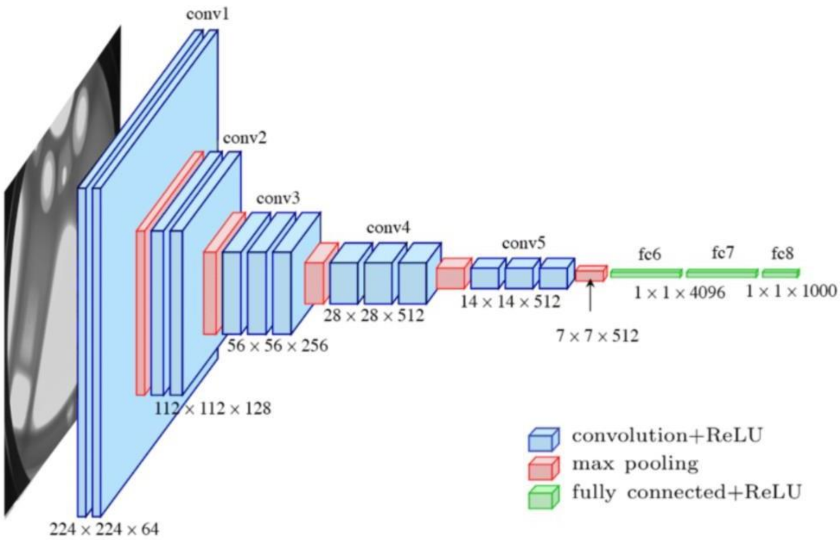


Fig 3.6. Architecture of CNN

3.6.2 DENSENET 121

Gao Huang et al. introduced the convolutional neural network architecture known as DenseNet-121 in 2017. It is a member of the DenseNet family of models, which encourages feature reuse to solve the issue of disappearing gradients in deep neural networks. Multiple dense blocks make up the DenseNet-121 architecture, and each block contains a set of convolutional layers that are densely connected to each other and to all layers that came before them. This means that each layer's output and each following layer's input in the block are concatenated. By doing this, DenseNet-121 encourages feature reuse and makes it possible for gradients to move across the network more effectively.

Various item recognition, segmentation, and other computer vision tasks including picture classification have been carried out using DenseNet-121. It has been used to categorize pictures of plant leaves or crops affected with various illnesses in agricultural disease prediction. One research, for instance, utilized DenseNet-121 to categorize pictures of maize leaves affected by four different diseases: grey leaves discoloration and northern leaf disease, common rust, and healthful. This work was published in the journal Computers and Electronics in Agriculture. On the test set, the authors outperformed other deep learning architectures and conventional machine learning algorithms, achieving an accuracy of 98.6%.

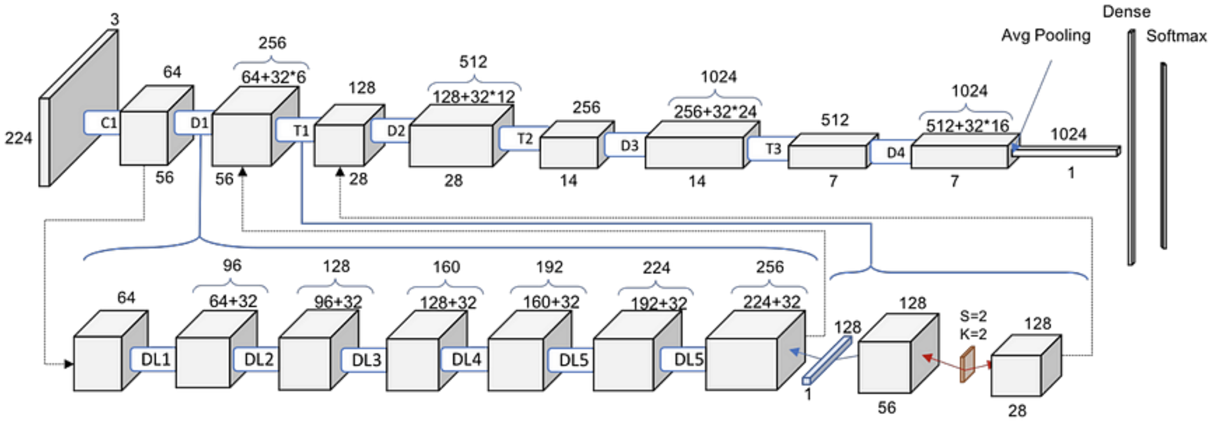


Fig 3.7. Dense Net layer and transition block

3.6.3 INCEPTION-V4

Christian Szegedy et al. presented the deep convolutional neural network architecture known as Inception-v4 in 2016. The Inception-v3 architecture, which was created for image classification problems, is an extension of that framework. Inception-v4 builds on Inception-v3's popularity and adds a number of updates to improve its functionality. The usage of a new block known as the "Inception-residual block" is the key innovation of Inception-v4. The characteristics of the original Inception block are combined with the ResNet architecture's residual connections in this block. The network may learn residual functions, or functions that are added to result of a single coating to create output of following coat, thanks to residual connections. This enables the network to avoid the vanishing gradient issue.

Inception-v4 has been used to categorise photos of plant leaves or crops affected with different illnesses in agricultural disease prediction. One research, for instance, employed Inception-v4 to categorise pictures of tomato leaves affected by four different diseases: early blight, late blight, leaf mould, and healthy. This work was published in the journal Computers and Electronics in Agriculture. The authors outperformed other deep learning architectures or conventional using machine learning by achieving a precision of 99.9% within criteria set.

Inception-v4 is utilised in a second study that was published in the journal Plant Methods to categorize photos of rice leaves affected by rice blast, leaf illness triggered by microbes, brown spot, and healthy leaves. Usefulness of Inception-v4 in predicting agricultural diseases was demonstrated by the authors, who attained a 97.2% accuracy on the test set.

By enabling early disease diagnosis, enhancing the sustainability and efficiency of farming operations, and emphasizing accessibility and inclusivity for farmers, the system has the potential to completely transform the agricultural sector. This enables farmers to discuss information based on their locality in forums that may be added to the program. The significance of inclusion and accessibility in technological solutions for the agriculture sector is emphasized by this part of the project.

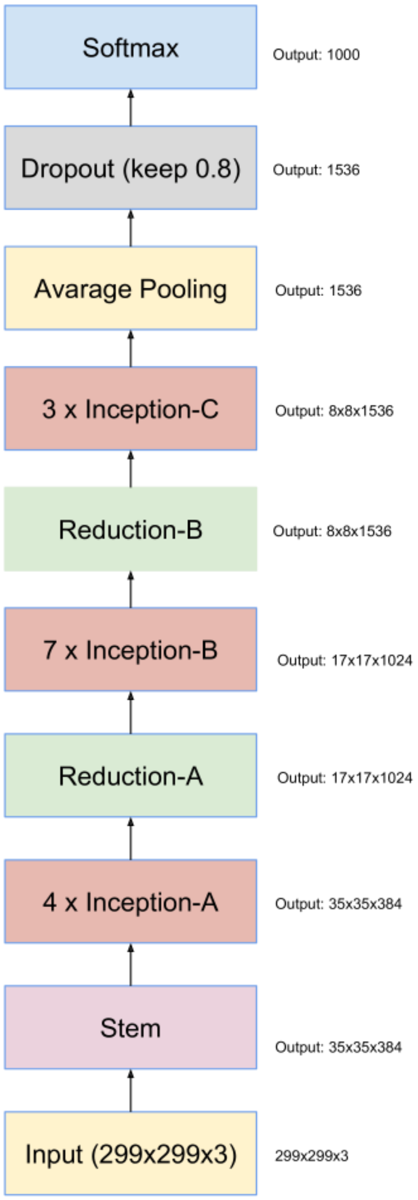


Fig 3.8. Architecture of Inception-V4 model

The addition of user-friendly interfaces that can be customized to suit regional languages and speech-to-text capabilities is another noteworthy feature. This enables farmers to discuss information based on their locality in forums that may be added to the program. The significance of inclusion and accessibility in technological solutions for the agriculture sector is emphasized by this part of the project. To sum up, the project "Crop Disease Prediction Using IoT and Deep Neural Networks" innovates by combining deep neural networks with IoT technologies to precisely and effectively identify crop illnesses.

3.7 Proposed DNN Architecture

Breast tumor classification using deep learning typically involves collecting and pre-processing the breast cancer imaging data. This involves cleaning the data, resizing the images, and normalizing the pixel values. Then increase the amount of data by performing random transformations on the existing data, such as rotations, flips, and zooms. Choose an appropriate deep-learning model for the classification task. Commonly used models for image classification include convolutional neural networks (DNNs), residual networks (ResNets), and inception networks. The knowledge gathered from previously trained models will then be applied to the new categorization assignment. This entails adjusting the previously trained model with fresh data. After transferring, we will use the pre-processed and enhanced data to train the deep learning model. Finally, we will examine the trained model's performance using a different set of test data.

Overall, the suggested architecture calls for feature extraction to be performed by a pre-trained DNN, and classification to be performed by a full-connected layer. A separate test set is utilized to assess algorithm's output following it has been trained using pre-processed and enhanced data.

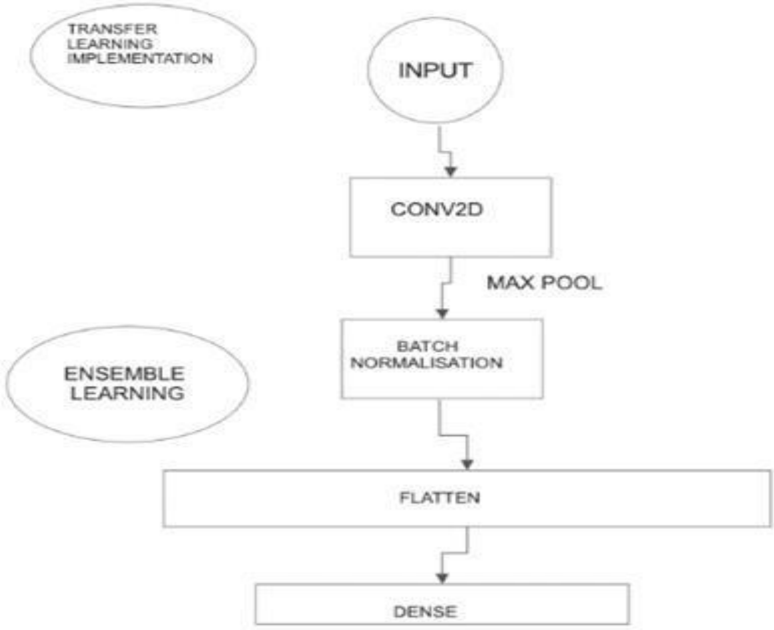


Figure 3.9. Proposed DNN Architecture Diagram

3.6.1 Module I (Data Collection)

Data gathering is a crucial part of the whole process for crop disease prediction utilizing IoT and deep neural networks. Data on the crops, their diseases, and their growing conditions are collected as part of this process. The accuracy of the prediction models can be considerably impacted by the type and volume of data gathered.

For the purpose of predicting agricultural diseases, a variety of data-collecting techniques are available, including crowdsourcing, ground-based sensors, and remote sensing. In order to collect information on crop health and growth patterns, remote sensing techniques such as satellite images are used. On the other hand, ground-based sensors are positioned right in the field to track environmental variables including humidity, temperature, and soil moisture.

- The use of IoT devices for data gathering for agricultural disease forecasting has become more popular in recent years.
- To gather information in actual time on elements like climate, moisture, soil moisture, other environmental variables, IoT sensors can be positioned directly on the crops or in the surrounding environment.
- The development of prediction models using this data can assist farmers in identifying and preventing crop illnesses.
- Another technique for gathering data that has become more popular recently is crowdsourcing.
- Data from a large number of people is gathered through crowdsourcing, typically using social media platforms or mobile apps.
- Using this information, a comprehensive database of crop diseases and their signs can be created.

It’s important to ensure that the collected data represents the population being studied and that it is of high quality. Additionally, obtaining informed consent and protecting patient privacy are critical considerations in the data collection process. In conclusion, data collecting for crop disease prediction utilising IoT and deep neural networks is a challenging process that needs careful planning and execution. It is possible to guarantee that the data gathered is reliable and pertinent by utilising a variety of data sources and cutting-edge technology, which can eventually result in more accurate crop disease prediction and prevention.

Gathering data for crop disease prediction utilising IoT and deep neural networks is a difficult process that needs careful planning and execution. Utilising a variety of data sources and cutting-edge technologies can aid in ensuring that the information gathered is accurate and pertinent, which can ultimately result in more accurate crop disease prediction and prevention.

Figure 3.6.1 Data Collection Representation

3.6.2 Module II (Feature Extraction)

Once you have the data, you need to preprocess it to prepare it for use in your model. To assess the effectiveness of your model, it's important to divide your data into three groups: a validation set, training set or test set. DDSM images vary in gray level due to differences in the scanners used to create them. Likewise, the images have varying optical densities and need to be normalized before the training process can begin.

Feature extraction is a crucial step that involves identifying appropriate components in the bare dataset to as worked as input on the deep learning model. Feature extraction is important because raw data such as medical images can be high dimensional and contain a lot of noise, making it difficult for deep learning models to learn meaningful patterns directly from the data. Feature extraction can be performed using different techniques, including:

- CNN : CNNs are frequently employed in picture-based classification applications for feature extraction. They are built to automatically recognize patterns in the input data and extract pertinent features.
- Pretrained Models: pretrained models such as VGG, ResNet, and Inception are trained on large image datasets and can be used for feature extraction by removing the last few layers of the model and using the output from the remaining layers as features.

- ❖ Handcrafted Features: Handcrafted features can be taken from raw data utilizing processing of images methods, such as edges detect or texture analysis.

Once the features are extracted, they are typically fed into a deep-learning model for classification. To know what to choose as feature extraction, the technique depends on the nature of the data, difficulty of classification work, or availability of labeled data for training.

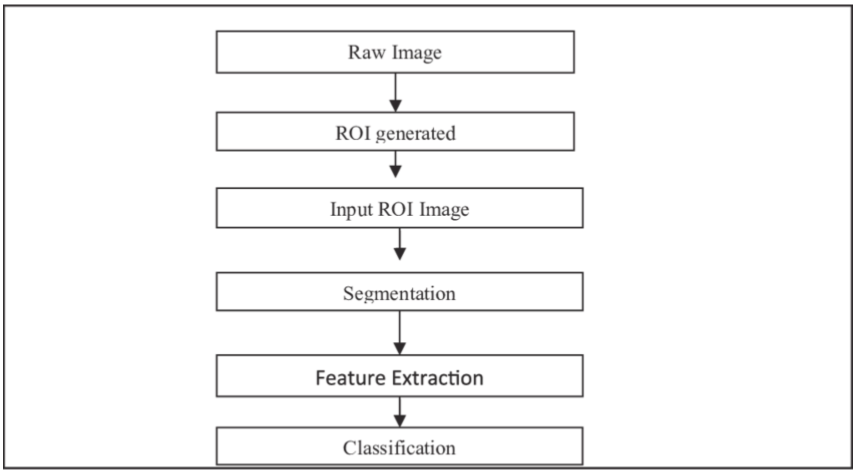


Figure 3.6.2 Feature Extraction Representation

3.6.3 Module III (Selecting and creating model)

The next is selecting the model based on the hyperparameter, type of layer, size, etc. The techniques above mentioned will help to design and build a model for better classification and prediction.

After feature extraction in Breast tumor classification using deep learning, following step is to choose suitable deep learning model of classification task. Choice of model is reliant on various substitutes, including nature of dataset, difficulty for classification work, number of classes, the accessibility of labelled data. Some common types of deep learning models that possibly used for breast tumor classification include:

- ❖ CNN: CNNs are commonly used for image-based classification tasks, including breast tumor classification. They are particularly effective in capturing spatial dependencies and extracting relevant characteristics of the input dataset.

- ❖ RNN: RNNs commonly employed for sequential data classification tasks, such as time-series analysis or natural language processing. They can also be used for breast tumor classification using features extracted from sequential data, such as gene expression data.
- ❖ Ensemble Models: Ensemble models combine multiple deep learning models to enhance accuracy or robustness of classification task. This can be particularly effective when dealing with complex or noisy data.

The selection of the appropriate deep learning model should be based on effectiveness of model throughout verification set the training procedure. It is crucial to make sure that the chosen model is capable of achieving high accuracy on the testing set and that it can generalize well to new data.

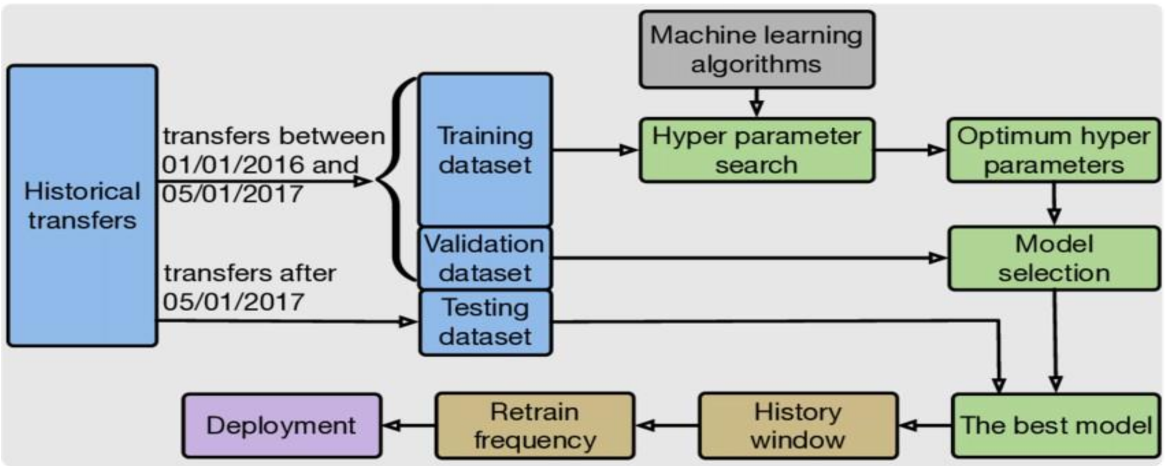


Figure 3.6.3: Model Selection Flowchart

3.6.4 Module IV (Multiple layers of classification)

This Module is the new thing that we added in this project. In these the original two classifications that are Benign and Malignant are further divided in four sub-layers each for all the classification of the tissue image that we extracted from the dataset.

Multiple layers of classification can be used after selecting a model in Breast tumor classification using deep learning. This involves using multiple deep learning models or layers to perform hierarchical classification of the input data. For example, in a breast tumor classification task, the first layer of classification may involve distinguishing between benign and malignant tumors. Multiple layers of classification can be implemented using various techniques, including:

- ❖ Stacked Ensemble Models: Stacked ensemble models combine multiple deep learning models, each trained on a different subset of features or input data, to perform hierarchical classification.
- ❖ Multi-Task Learning: A single deep learning model is trained to carry out several categorization tasks at once using multi-task learning. In light of breast tumor classification, this could involve training a single model to distinguish between benign and malignant tumors as well as classify malignant tumors into subtypes.
- ❖ Transfer Learning: Learning Transfer entails utilising a pretrained deep learning model as feature extractor for fresh classification work. In light of classification, this could involve using a pretrained model trained on a similar classification task, such as lung tumor classification, to extract relevant features for the breast tumor classification task.

The choice of multiple layers of classification technique depends on difficulty of classification work, number of classes, accessibility of labelled dataset for training. It is important to evaluate the performance of the multiple layers of classification approach on the validation and testing sets to ensure that it is effective in improving the accuracy of the classification task.

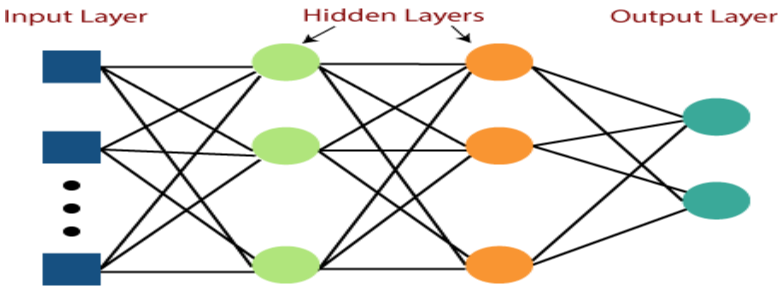


Figure 3.6.4 Multiple Layers of Classification

3.6.5 Module V (Training and Evaluation)

Upon the completion of the model architecture and compilation, you can start training the model with training set. Evaluation of its effectiveness is based on validation dataset. This allows you to tune the hyperparameters and make any necessary modifications to the model architecture. After implementing multiple layers of classification in Breast tumor classification using deep learning, the following step is training and evaluate our model. Training the model involves using a labelled dataset to optimize the parameters of utilising a deep learning model suitable optimization algorithm, such as Stochastic Gradient Descent (SGD). The dataset is typically split into training, validation, and testing sets. The validation set is used to adjust the hyperparameters and avoid overfitting, the training set is used to update the model parameters, and the testing set is used to assess the trained model's performance. Evaluation of trained model involves calculating performance indicators such as accuracy, precision, recall, F1 score, and ROC curve on the testing set.

These metrics provide an estimate of how well the model can classify new, unseen data. In the context of multiple layers of classification in Breast tumor classification, the evaluation process can be performed separately for each layer of classification. For example, the accuracy of the first layer of classification can be evaluated by comparing the predicted labels of the benign and malignant tumors to their ground truth labels. Similarly, the accuracy of the second layer of classification can be evaluated by comparing the predicted labels of the malignant tumor subtypes to their ground truth labels. It is important to ensure that the model(s) perform well on both the validation and testing sets and that they can generalize well to new, unseen data. If the model(s) do not perform well, it may be necessary to retrain the model(s) with different hyperparameters or adjust the architecture of the deep learning model(s).

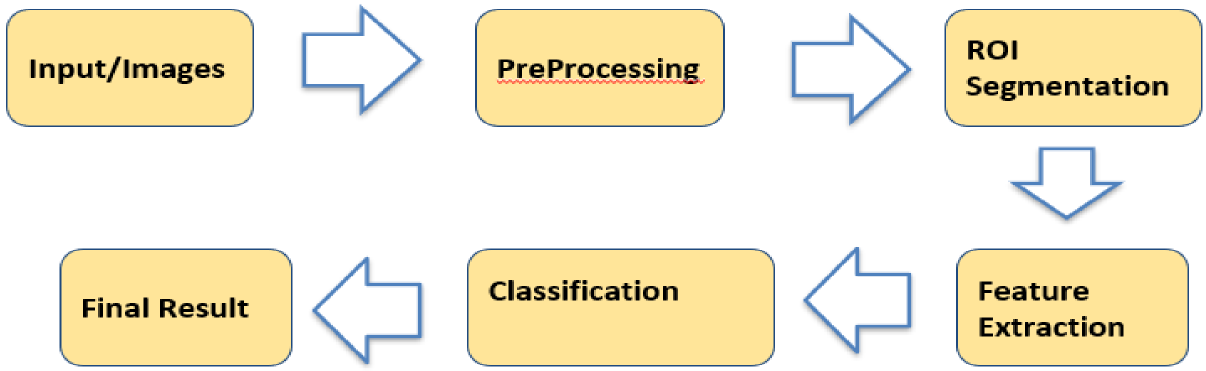


Figure 3.6.5: System Architecture Diagram

After implementing multiple layers of classification using deep learning, the following step is training and evaluate our model. Training the model involves using a labelled dataset to optimize the parameters of the deep learning model using a suitable optimization algorithm, such as Stochastic Gradient Descent (SGD). In most cases, the dataset is divided into training, validation, and testing sets. The validation set is used to adjust the hyperparameters and prevent overfitting, the training set is used to update the model parameters, and the testing set is used to assess the trained model's performance.

Utilising a deep learning model that has already been trained is known as transfer learning model as feature extractor for a new classification task. In light of classification, this could involve using a previously trained model skilled in a comparable classification task, such as classification, to extract relevant features for the classification task.

CHAPTER 4

RESULT AND DISCUSSION

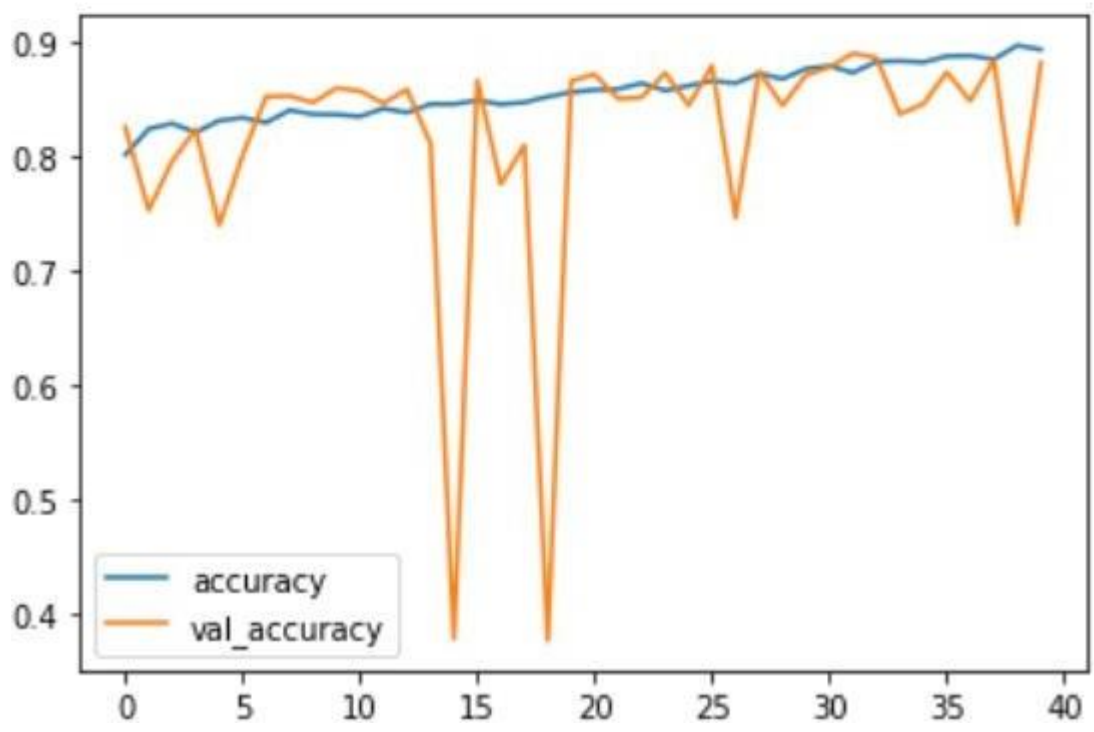


Figure. 4.1 Data Accuracy Graph

The amount and consistency of the practise data, the complexity of prototype architecture, or choice of hyperparameters can all affect how accurate a deep learning model is at classifying breast tumors. The dataset must be properly chosen and preprocessed, the model architecture must be adequate, and the model hyperparameters must be optimized using methods like cross-validation and grid search. In order to make sure that the model generalizes effectively to previously unexplored data, it is also critical to assess the model's performance on a validation dataset.

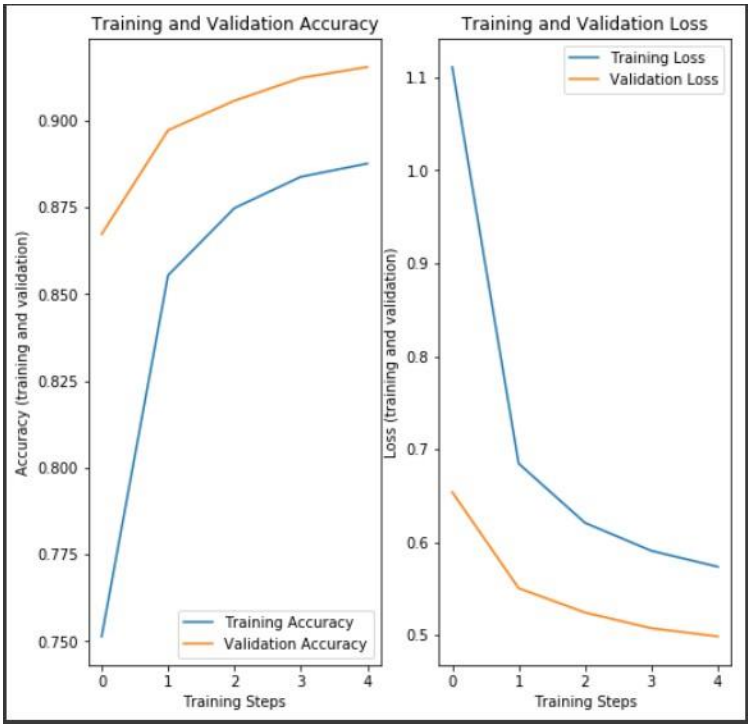


Fig 4.2. Plot for training and validation

Data gathering is essential for building precise and dependable models in the context of crop disease prediction utilising IoT and deep neural networks. Deep learning algorithms may be taught on big picture datasets to find patterns and characteristics that may help distinguish between healthy and unhealthy plants, similar to how breast cancer classification algorithms are learned. In order to provide a thorough study of plant health, these algorithms can analyze the whole picture, including details that might not be visible to the unaided eye. Images of plants and leaves are taken from a variety of locations, including farms, plantations, and research facilities, as part of the data collection process for crop disease prediction. The type of plant, the portion of the plant, and other pertinent metadata must be appropriately tagged in these photos.

In order to increase model accuracy, additional information may be gathered in addition to photos, such as weather data, soil quality data, and other environmental parameters. IoT gadgets like weather sensors, moisture sensors, and other environmental monitoring tools may be used to do this. In order to train precise and dependable models, data collection for crop disease prediction utilizing IoT and deep neural networks entails the capture, labeling, and integration of many forms of data. Data collection is a crucial stage in the creation of efficient systems for the prediction of crop diseases since the quality and amount of the data gathered directly affects the performance of the models.

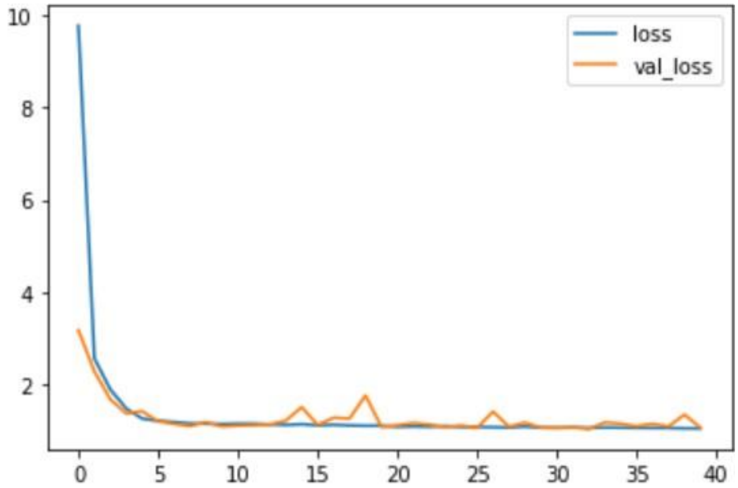


Figure. 4.3 Error (Loss) Graph

During the training of a deep learning model for breast tumor classification, the model undergoes multiple iterations or epochs, where it learns to identify patterns and features in the images. In each epoch, the model updates its weights the discrepancy between the anticipated and actual values, or loss function, must be reduced in order to actual label of the images. To evaluate the model's performance during training, it is crucial to monitor both the inaccuracy in training and the validation error. The learning blunder measures difference between the predicted and actual labels on the training data, while the validation error measures the error on a separate set of data that the model has not seen before. If the low training error, and the validation high error, it that the model is too closely fitting the training set of data and may not generalize well to new data. Conversely, if both the training error and validation error are high, it implies that the model is underfitting and not learning enough based on the data. To enhance model's efficiency, the goal is to minimize the validation error while avoiding overfitting. This can be accomplished by changing the model's hyperparameters, including the learning rate, the number of layers, and the amount of neurons, and through regularization techniques like dropping out and shed pounds. By monitoring training and validation errors during training, healthcare providers can ensure that the deep learning model is accurately and reliably classifying breast tumors, leading to better treatment.

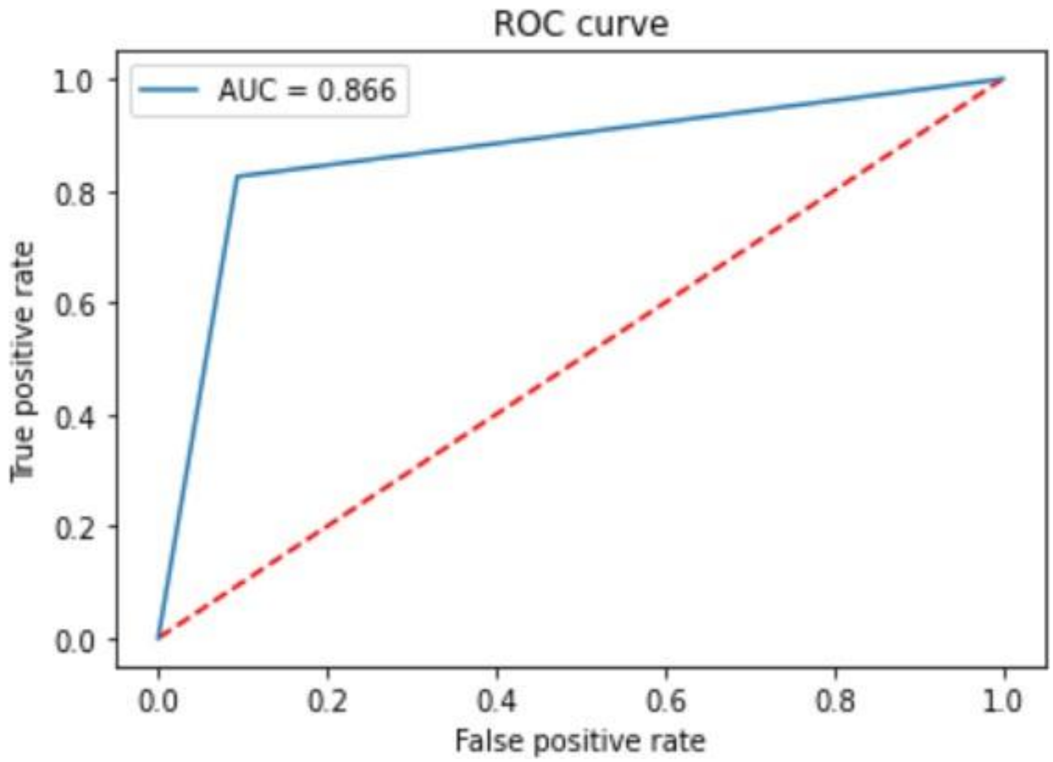


Figure. 4.4 ROC Performance Curve

Crop disease prediction using deep learning can be evaluated using a ROC bend, which shows the performance of the binary classifier as the discrimination threshold is changed graphically. To generate a ROC curve, the deep learning model is first built from a dataset of breast tumor pictures with labels indicating whether each tumor is benign or malignant. Then, the model is applied to a separate validation dataset, and the predicted probabilities of each tumor being malignant are calculated. A range of classification thresholds is chosen, and the true positive rate and false positive rate is calculated for each threshold. The TPR represents the proportion of malignant tumors correctly classified as such, while the FPR represents the proportion of benign tumors incorrectly classified as malignant. These values are plotted on a graph where the x-axis is the FPR and the y-axis is the TPR, and the ROC curve for the deep learning model is plotted on top of a diagonal line that represents the ROC curve for a random classifier. The AUC of the ROC curve can be used as a measure of the overall performance of the deep learning model, with 1.0 AUC indicating perfection classification performance with an AUC is 0.5 indicating random guessing. By evaluating the ROC curve and AUC, healthcare providers can assess the accuracy and reliability of the deep learning model in breast tumor classification, leading to better treatment decisions and improved patient outcomes.

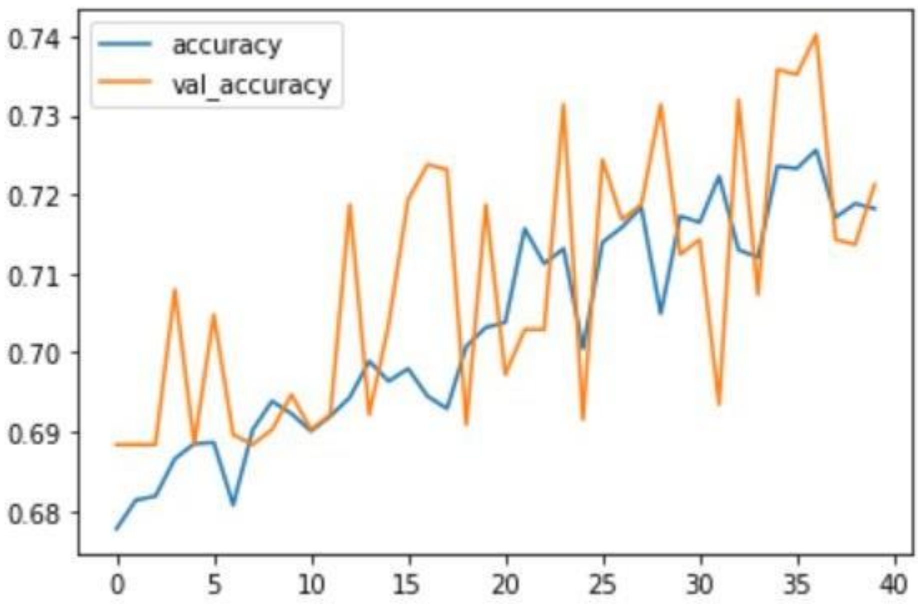


Figure. 4.5 Validation Accuracy Graph

Validation accuracy is a key criterion in assessing a deep learning model for classifying breast tumours. A different subset of data, known as the validation set, is used to assess the model's performance after it has been trained on the first sample of data. Applying the trained model to the validation set and comparing the predicted labels to the actual labels yields the validation accuracy. However, evaluating a deep learning model's performance solely on validation accuracy is insufficient. Other measures like accuracy, recall, F1 score, and ROC curve analysis should also be taken into account to make sure the model is effective in making correct predictions. Multiple metrics are required to provide a thorough evaluation because a high validation accuracy does not guarantee the model's accuracy and reliability in classifying breast tumours.

Utilising a variety of metrics is crucial when assessing a deep learning model's performance for classifying breast tumours in order to guarantee the model's accuracy and dependability. One significant and frequently employed statistic is the validity accuracy.

CHAPTER 5

CHALLENGES AND LIMITATIONS

There are a number of difficulties and restrictions with IoT-based deep neural network crop disease prediction. For precise prediction, data availability and quality are essential, yet gathering and labelling huge datasets can be time-consuming and expensive. Additionally, worries about data privacy may restrict the sharing of data between various stakeholders. The choice and optimisation of deep learning models presents another difficulty because they demand a large amount of processing power and knowledge. Additionally, signal interference from IoT devices used for data collection could have an impact on the data's accuracy. Last but not least, the system's deployment in actual agricultural settings may encounter technical and practical difficulties related to maintenance, compatibility, and scalability.

The collecting and annotation of data provide one of the largest obstacles. It can take lots of time or resources to gather enough dataset to train deep neural networks, and labelling the data can be arbitrary and prone to mistakes. The accuracy and generalizability of the models may also be constrained by the data's potential bias towards particular geographic areas or crop varieties. The requirement for high-quality sensors and IoT devices to gather precise data on crop conditions is another difficulty. For small-scale farmers or those with limited means, these devices' high cost and frequent maintenance requirements might be a deterrent.

- Limited Availability of Data

The scarcity of high-quality data is one of the main obstacles to accurate crop disease prediction. Large datasets can be time- and money-consuming to gather and label, and the information may also be inaccurate or incomplete. The restricted data availability is one of the primary obstacles to crop disease prediction utilising IoT and deep neural networks. Large-scale, high-quality data collection on agricultural diseases can be challenging and time-consuming, especially in places with little resources. Collecting representative data for all crops and diseases is also challenging because some crops are more prone to specific diseases than others. As a result, datasets may become unbalanced, which could harm the effectiveness of deep learning models.

- Lack of Standardization

It is challenging to compare findings between research and geographical areas since crop disease data collecting and reporting are not standardised. The absence of standards in data collection, annotation, and labelling is another difficulty in crop disease prediction utilising IoT and deep neural networks. For deep learning models to be accurate and reliable, the quality of the data used to train and test them is essential. Deep learning models must be trained to be able to generalise well to new data, however lack of standardisation in data gathering can result in inconsistencies in the dataset. Additionally, the absence of standardised labelling techniques may lead to inaccurate annotations and mislabeling, which may impair the accuracy of deep learning models. By creating standardised methods and instructions for data collection, annotation, and labelling in crop disease prediction, this problem can be overcome.

- Technical Challenges

IoT system implementation might be technically difficult, particularly in distant locations with poor connectivity or power supplies. Deep neural network training and deployment can demand a lot of computer power, which may not be easily accessible to farmers or smaller businesses. Using IoT and deep neural networks to detect agricultural diseases presents a number of technological obstacles for academics. The necessity for quick and precise data collection is one of the main challenges. Due to problems like signal interference and poor connectivity, gathering data from IoT devices like sensors and cameras can be difficult. It can also be difficult to process the vast amounts of data produced by IoT devices, which calls for the utilisation of high-performance computing resources.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENTS

The use of automation and IoT technology in agricultural practices has revolutionized the way farmers approach crop disease detection and prevention. The ability to remotely monitor fields and quickly identify and treat diseased crops using deep learning models has led to significant improvements in crop yield and quality. As technology continues to evolve, the use of drone-mounted cameras and improved lighting conditions will further enhance the accuracy and efficiency of this approach. The benefits of using IoT and deep learning in agriculture extend beyond just disease detection and prevention. Cost control, waste reduction, process automation, improved product quality, and increased output are all potential benefits that farmers can enjoy by adopting this approach. Moreover, this approach is especially helpful for farmers who lack access to professional knowledge and assistance, making it a cost-effective and scalable solution. Optimizing the method used to transport data to the cloud and using Python programs for cloud analytics are critical aspects of this approach. Storing, managing, and retrieving information from the database is also a crucial aspect of this system. Neural networks are used to instruct the system to understand diseases and provide treatment recommendations, which can be communicated to farmers.

In conclusion, the integration of IoT and deep learning in agriculture has revolutionized the way farmers approach crop disease detection and prevention. This approach has many benefits, including cost control, waste reduction, process automation, improved product quality, and increased output. As technology continues to evolve, it is expected that the accuracy and efficiency of this approach will continue to improve, leading to even more benefits for farmers and the wider community.

Using IoT and deep neural networks to detect crop diseases, lack of standardisation is a significant barrier. Adopting open data standards will make it easier for platforms and systems to share data, which will help address this problem. Decision trees and rule-based models are two explainable AI strategies that can aid increase user acceptance of crop disease prediction systems by giving clear and comprehensible justifications for the predictions provided by the system.

REFERENCES

[1] HemantKumar Wani, Nilima Ashtankar "An Appropriate Model Predicting Pest/Diseases of Crops Using Machine Learning Algorithms", International Conference on Advanced Computing and Communication Systems (ICACCS-2017), Jan 06-07, 2017.

[2] Srishti Rawal, "IOT based Smart Irrigation System", International Journal of Computer Applications (0975 – 8887) Volume 159 – No 8, February 2017.

[3] Neha Kurale, Madhav Vaidya, "Classification of Leaf Disease Using Texture Feature and Neural Network Classifier", 2018 International Conference on Inventive Research in Computing Applications (ICIRCA).

[4] Aishwarya Kagalkar, "Smart Irrigation System", IJERT, Vol. 6 Issue 05, May 2017.

[5] Aman Kumar, Ajit Kumar, "Smart Irrigation System using IoT:SIS", IJERT, Vol. 6 Issue 06, June 2017

[6] Mekala.S, Sarala.S, Divya.S, "Microcontroller based Smart Irrigation System and Weather Forecasting using Anemometer", Volume 118 No. 20 2018, 2931-2936, IJPAM, 2018.

[7] Nikesh Gondchawar, Dr. R. S. Kawitkar, "IoT based Smart Agriculture", IJARCCE, Vol. 5, Issue 6, June 2016

[8] Zarreen Naowal Reza, Faiza Nuzhat, Nuzhat Ashraf Mahsa, Md. Haider Ali , "Detecting jute plant disease using image processing and machine learning", 2016 3rd International Conference on Electrical Engineering and Information Communication Technology (ICEEICT).

[9] Meliks Ardogan, Adam Tuncer, Yunnus Ozen, "Plant Leaf Disease Detection and Classification Based on CNN with LVQ Algorithm", 2018 3rd International Conference on Computer Science and Engineering (UBMK).

[10] Jie You, Wei Liu, Joonwhoan Lee, "A DNN-based semantic segmentation for detecting weed and crop", Computers and Electronics in Agriculture, Vol. 178, November 2020.

[11] Yuzhen Lu, Sierra Young, "A survey of public datasets for computer vision tasks in precision agriculture, Computers and Electronics in Agriculture,", Volume 178, 2020, 105760, ISSN 0168- 1699.

[12] S. Ramesh and Bharghava Rajaram," IoT Based Crop Disease Identification System Using Optimization Techniques", ARPN Journal of Engineering and Applied Sciences vol. 13, no. 4, february 2019.

[13] Pallavi. S. Marathe, "Plant Disease Detection using Digital Image Processing and GSM", International Journal of Engineering Science and Computing, April 2017, pp. 10513-15.

[14] R.Rajmohan, M.Pajany, Smart paddy crop disease identification and management using deep convolution neural network & svm classifier, International journal of pure and applied mathematics, vol 118, no 5, pp. 255-264, 2017.

[15] A. A. Joshi and B. D. Jadhav. 2016. Monitoring and controlling rice diseases using Image processing techniques. International Conference on Computing. Analytics and Security Trends (CAST), Pune. pp. 471-476.

[16] S. Mutalib, M. H. Abdullah, S. Abdul-Rahman and Z.A. Aziz. 2016. A brief study on paddy applications with image processing and proposed architecture. IEEE Conference on Systems, Process and Control (ICSPC), Melaka, Malaysia. pp. 124-129.

[17] Jagadeesh D. Pujari, Rajesh Yakkundimath, Abdulmunaf S. Byadgi. 2015. Image Processing

[19] Mohanty SP, Hughes DP and Salathé M (2016) Using Deep Learning for Image-Based Plant Disease Detection. Front. Plant Sci. 7:1419. doi: 10.3389/fpls.2016.01419

[20] R. M. Haralick, K. Shanmugam and I. Dinstein, "Textural Features for Image Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, pp. 610-621, Nov. 1973, doi: 10.1109/TSMC.1973.4309314.

[21] Kulkarni, P. et al. (2021) Plant disease detection using image processing and machine learning, arXiv.org. Available at: https://arxiv.org/abs/2106.10698.

[22] Abhirami, K., Praba, G.C. and Lakshmi, R.S. (2021) IOT based Paddy Crop Disease Identification and prevention system using deep neural networks and image processing, International Journal of Engineering Research & Technology. IJERT-International Journal of Engineering Research & Technology. Available at: https://www.ijert.org/iot-based-paddy-crop-disease-identification-and-prevention-system-using-deep-neural-networks-and-image-processing.

[18] Abhirami, K., Praba, G.C. and Lakshmi, R.S. (2021) IOT based Paddy Crop Disease Identification and prevention system using deep neural networks and image processing, International Journal of

Engineering Research & Technology. IJERT-International Journal of Engineering Research & Technology. Available at: https://www.ijert.org/iot-based-paddy-crop-disease-identification-and-prevention-system-using-deep-neural-networks-and-image-processing.

[19] Kulkarni, P. et al. (2021) Plant disease detection using image processing and machine learning, arXiv.org. Available at: https://arxiv.org/abs/2106.10698.

[20] Mohanty SP, Hughes DP and Salathé M (2016) Using Deep Learning for Image-Based Plant Disease Detection. Front. Plant Sci. 7:1419. doi: 10.3389/fpls.2016.01419

[21] R. M. Haralick, K. Shanmugam and I. Dinstein, "Textural Features for Image Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, pp. 610-621, Nov. 1973, doi: 10.1109/TSMC.1973.4309314.

[22] Jie You, Wei Liu, Joonwhoan Lee, “A DNN-based semantic segmentation for detecting weed and crop”, Computers and Electronics in Agriculture, Vol. 178, November 2020.

APPENDIX I

This part of the report contains the information about the various packages and the languages we used to execute our code to get the desired output.

We executed our code in python language, in google colaboratory environment.

This project was developed using Python, a general-purpose, interpreted, interactive, object-oriented, and high-level programming language. It offers readable and transparent code. Despite being extremely complex with flexible procedures, AI and ML algorithms can help developers create reliable machine intelligent systems when they are implemented in Python. The Python packages used in our project are listed below:

- **Pandas:** The package is made in Python and offers effective, adaptable data structures for "labelled" or "relational" data that are simple to use. Its main objective is to establish itself as a key building block for Python-based data analysis by providing a high-level, user-friendly methodology. The goal of the package is to become the most robust and flexible open-source tool for data analysis and manipulation in any programming language.

- **Numpy:** The library is a Python module that offers a wide range of derived objects, such as a multidimensional array object and masked arrays and matrices, along with a number of functions that enable quick array operations, including fundamental statistical operations, discrete Fourier transforms, fundamental linear algebra, and random simulation. This module forms the basis of many scientific computing projects and is crucial for scientific computing in Python.

- **OpenCV-python:** A complete infrastructure for computer vision applications is provided by the free computer vision and machine learning software library known as OpenCV. It offers a big library of over 2500 optimised algorithms that span a wide variety of conventional and cutting-edge computer vision and machine learning approaches in order to make it easier to incorporate artificial intelligence into products. These methods may be applied to a number of tasks, including object recognition, tracking camera motions, face detection and recognition, picture stitching, and object removal from films. Additionally, they have the ability to create 3D models of objects, extract 3D point clouds from stereo cameras, categorise human actions in videos, and locate related images in an image database.

- **Imgaug:** It serves as a library for experiments in machine learning that employ photo augmentation. It has a simple yet effective stochastic interface, supports a wide variety of augmentation techniques, makes it simple to combine them and execute them in random order or on multiple CPU cores, and can enhance not only images but also keypoints/landmarks, bounding boxes, heatmaps, and segmentation maps.

- **PIL:** The Python Imaging Library was made by Fredrik Lundh and other contributions. It improves the image processing capabilities of the Python interpreter and is designed for easy utilisation of data stored in a few basic pixel formats.

- **Matplotlib:** For creating interactive, lively, and static in Python, this is a thorough library. It can construct interactive figures with zoom, pan, update, and customizable visual style and layout, export to a variety of file formats, and be incorporated in JupyterLab and graphical user interfaces. It can also produce plots suitable for publishing.

- **Keras:** It is a Python interface for an open-source software library that supports artificial neural networks. The TensorFlow library interface is provided by Keras. TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML were just a few of the backends that Keras supported up until version 2.3.

- **Glob:** It is a Python standard library module that offers a function for locating all the pathnames that fit a given pattern in accordance with the Unix shell's rules, however outcomes vary depending on the platform. The Unix shell uses a pattern matching algorithm like this one. For discovering files and directories that match a specific pattern, use the glob module. It can be used, for instance, to locate all files in a directory that have a specific file extension.

- **Json:** JSON (JavaScript Object Notation) is a straightforward data format for both developers and users, humans and machines to read, write, parse, and produce. The json module in Python provides built-in functionality for interacting with JSON data. The json module offers the json.dumps() and json.loads() functions for encoding and decoding Python objects as JSON strings, respectively. For further finer-grained control over the encoding and decoding process, it also offers additional features.

- **Os:** It is a module in Python's standard library that gives access to specifics of the operating system like reading from or writing to the file system, making and deleting files and directories, and working with environment variables. A portable method of dealing with the file system and the operating system is offered by the os module. The os module also offers a variety of other helpful tools for interacting with the operating system and file system, including operations for modifying environment variables, executing commands in a shell, and handling permissions. The os module can be imported with the import os command, and any of its functions can then be used as necessary in your code. When utilising certain of the os functions, it's crucial to be aware of any platform-specific behaviour because operating systems may cause these functions to act differently.

- **Imutils:** It is a package that offers a collection of useful functions to make working with OpenCV, a well-known Python computer vision library, simpler. The imutils package includes tools for working with colours, displaying images, rotating, cropping, and resizing images. Imutils also offers tools for working with colours, such as rgb to hsv and hsv to rgb, as well as tools for showing images using imshow.

❖ **TensorFlow:** It is a freely available machine learning and artificial intelligence computer toolkit and free. Although it can be applied to many different tasks, deep neural network training and inference are given special attention. The Google Brain team created TensorFlow for use in internal Google research and production. Many programming languages, including Python, Javascript, C++, and Java, can use TensorFlow. Researchers can advance the state-of-the-art in ML thanks to its extensive, adaptable ecosystem of tools, libraries, and community resources, while developers can simply create and deploy ML-powered apps.

❖ **Paths:** A path in Python denotes a file or directory's position on the file system. Python has a number of path-related modules, including os.path, pathlib, and glob. The os.path module, which is a component of the os module, offers platforms-independent utilities for working with file paths. Path manipulation tools including join, basename, dirname, abspath, split, splitext, and more are available in the os.path module.

❖ **Sklearn:** Python programmers have access to a free machine learning package. Numerous algorithms, including support-vector machines, random forests, gradient boosting, k-means, and DBSCAN, are provided for classification, regression, and clustering. It is also intended to work with Python's NumPy and SciPy scientific and mathematical libraries.

❖ **Scipy:** It is an open-source, cost-free Python library that is employed in technical and scientific computing. It includes modules for signal and image processing, interpolation, integration, special functions, and optimization. SciPy uses a multidimensional array as its fundamental data structure, which is made available through the NumPy module.

❖ **Functools:** It is a module in the standard library of Python that offers a number of higher-order functions that can be used to modify and combine existing functions. Higher-order functions can either output another function or take one or more additional functions as inputs. It provides several features out of which a few a mentioned below:

- Partial- By using this function, a new function that has some of its arguments already filled in can be made from an existing function. When you need to frequently call a function with the same set of arguments but different values for the other arguments, this can be helpful.

- Reduce- With this function, you can condense a list of values into a single value. Once a function and a sequence have been entered, the function is repeatedly applied to pairs of values from the series until only one value is left.
- Cached_Property- A class instance's property result can be cached with this decorator. When calculating a property's value is expensive but the value is not likely to change frequently, this can be helpful.

❖ **Itertools:** It is a different module that offers a set of tools for working with iterable objects in the Python standard library. Lists, tuples, and dictionaries are examples of iterable objects, which are things that may be repeated.

APPENDIX II

This section of the report contains the code we built for our project.

Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

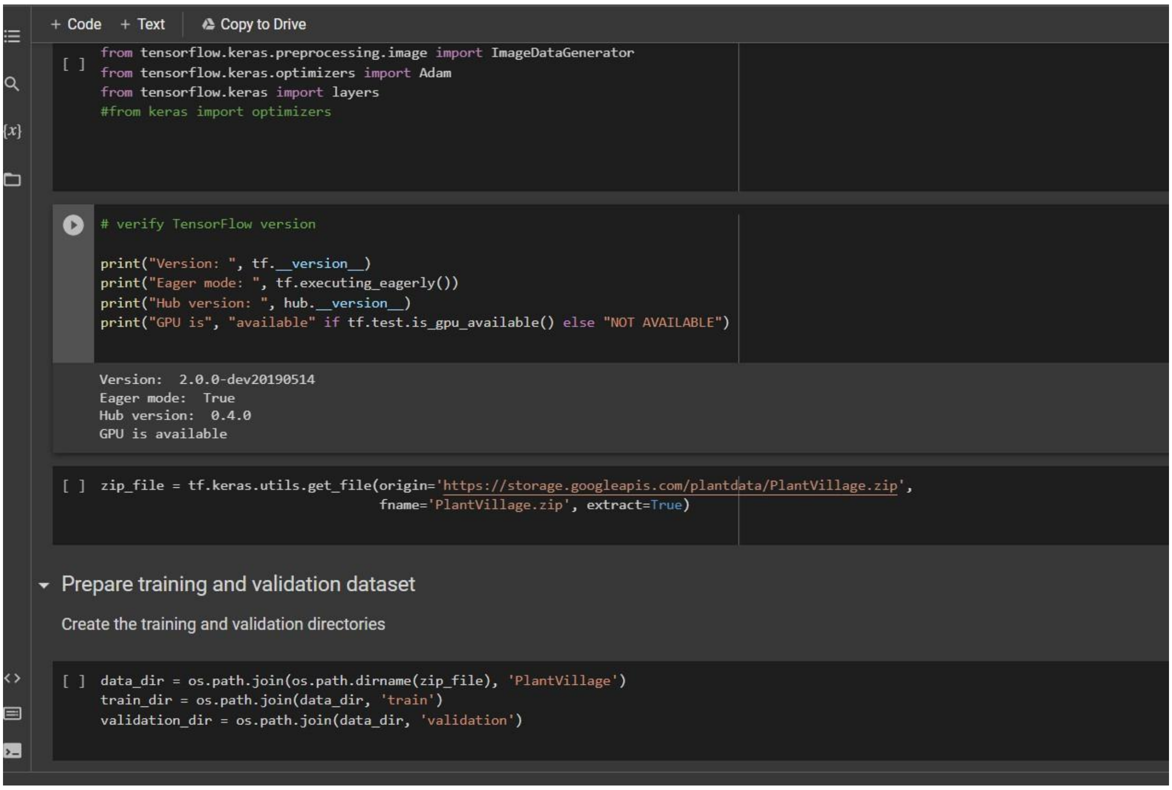
+ Code + Text Copy to Drive

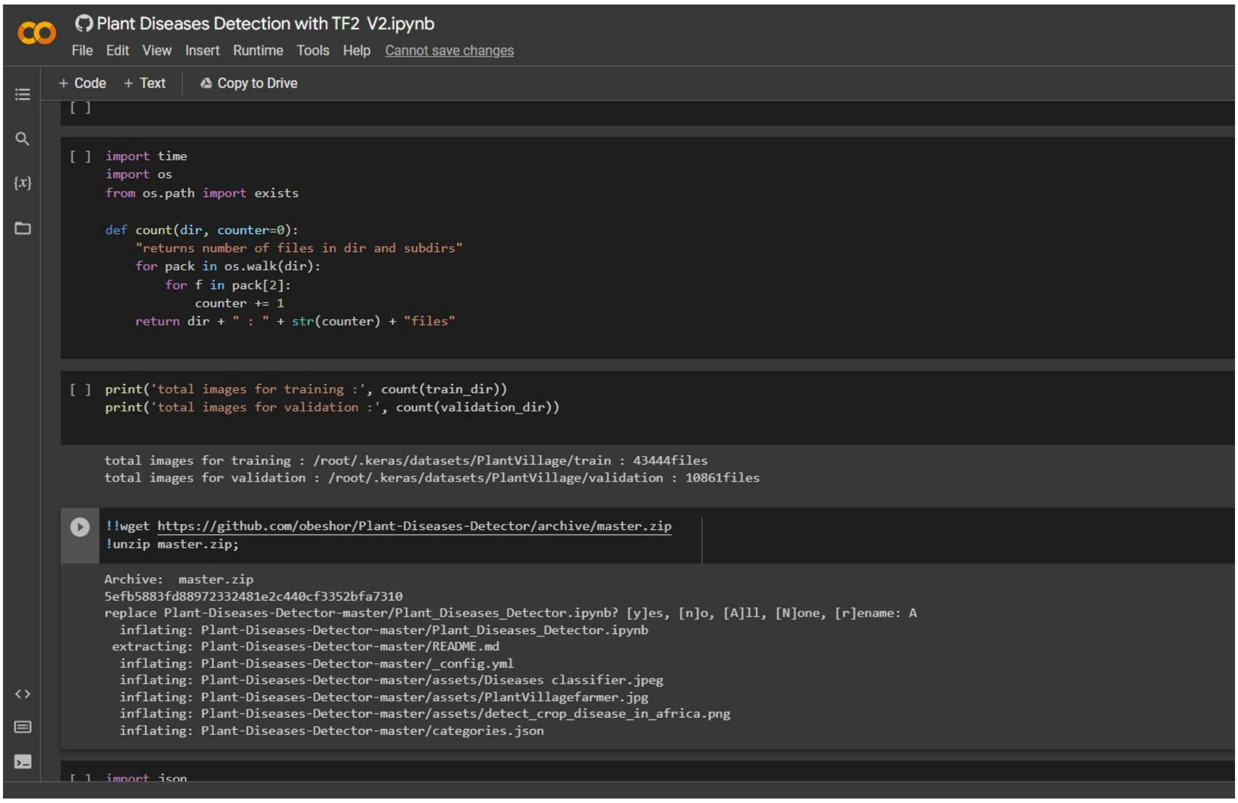
Importing the Librairies

```
# Install nightly package for some functionalities that aren't in alpha
!pip install tf-nightly-gpu-2.0-preview

# Install TF Hub for TF2
!pip install 'tensorflow-hub == 0.4'
```

```
from __future__ import absolute_import, division, print_function, unicode_literals


import tensorflow as tf
#tf.logging.set_verbosity(tf.logging.ERROR)
#tf.enable_eager_execution()

import tensorflow_hub as hub
import os
from tensorflow.keras.layers import Dense, Flatten, Conv2D
from tensorflow.keras import Model
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import layers
#from keras import optimizers
```

```
# verify TensorFlow version

print("Version: ", tf.__version__)
print("Eager mode: ", tf.executing_eagerly())
print("Hub version: ", hub.__version__)
```


+ Code + Text
Copy to Drive
[ ]
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import layers
#from keras import optimizers
# verify TensorFlow version
print("Version: ", tf.__version__)
print("Eager mode: ", tf.executing_eagerly())
print("Hub version: ", hub.__version__)
print("GPU is", "available" if tf.test.is_gpu_available() else "NOT AVAILABLE")
Version: 2.0.0-dev20190514
Eager mode: True
Hub version: 0.4.0
GPU is available
[ ]
zip_file = tf.keras.utils.get_file(origin='https://storage.googleapis.com/plantdata/PlantVillage.zip',
fname='PlantVillage.zip', extract=True)
Prepare training and validation dataset
Create the training and validation directories
[ ]
data_dir = os.path.join(os.path.dirname(zip_file), 'PlantVillage')
train_dir = os.path.join(data_dir, 'train')
validation_dir = os.path.join(data_dir, 'validation')


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text
Copy to Drive
[ ]
import time
import os
from os.path import exists
def count(dir, counter=0):
"returns number of files in dir and subdirs"
for pack in os.walk(dir):
for f in pack[2]:
counter += 1
return dir + " : " + str(counter) + "files"
[ ]
print('total images for training :', count(train_dir))
print('total images for validation :', count(validation_dir))
total images for training : /root/.keras/datasets/PlantVillage/train : 43444files
total images for validation : /root/.keras/datasets/PlantVillage/validation : 10861files
!!wget https://github.com/obeshor/Plant-Diseases-Detector/archive/master.zip
!unzip master.zip;
Archive: master.zip
5efb5883fd88972332481e2c440cf3352bfa7310
replace Plant-Diseases-Detector-master/Plant_Diseases_Detector.ipynb? [y]es, [n]o, [A]ll, [N]one, [r]ename: A
inflating: Plant-Diseases-Detector-master/Plant_Diseases_Detector.ipynb
extracting: Plant-Diseases-Detector-master/README.md
inflating: Plant-Diseases-Detector-master/_config.yml
inflating: Plant-Diseases-Detector-master/assets/Diseases classifier.jpeg
inflating: Plant-Diseases-Detector-master/assets/PlantVillagefarmer.jpg
inflating: Plant-Diseases-Detector-master/assets/detect_crop_disease_in_africa.png
inflating: Plant-Diseases-Detector-master/categories.json
[ ]
import json

Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

Now that you've trained the model, export it as a saved model

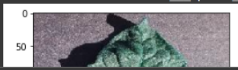
```
import time
t = time.time()

export_path = "/tmp/saved_models/{}".format(int(t))
tf.keras.experimental.export_saved_model(model, export_path)

export_path
```

```
# Now confirm that we can reload it, and it still gives the same results
reloaded = tf.keras.experimental.load_from_saved_model(export_path, custom_objects={'KerasLayer':hub.KerasLayer})
```

```
def predict_reload(image):
    probabilities = reloaded.predict(np.asarray([img]))[0]
    class_idx = np.argmax(probabilities)

    return {classes[class_idx]: probabilities[class_idx]}
```

```
for idx, filename in enumerate(random.sample(validation_generator.filenames, 2)):
    print("SOURCE: class: %s, file: %s" % (os.path.split(filename)[0], filename))

    img = load_image(filename)
    prediction = predict_reload(img)
    print("PREDICTED: class: %s, confidence: %f" % (list(prediction.keys())[0], list(prediction.values())[0]))
    plt.imshow(img)
    plt.figure(idx)
    plt.show()
```

```
SOURCE: class: Tomato___healthy, file: Tomato___healthy/b65ccffe-a2fc-44d1-b56f-c8e97db5232e___RS_HL 0120.JPG
PREDICTED: class: Tomato___Spider_mites Two-spotted_spider_mite, confidence: 0.607259
```

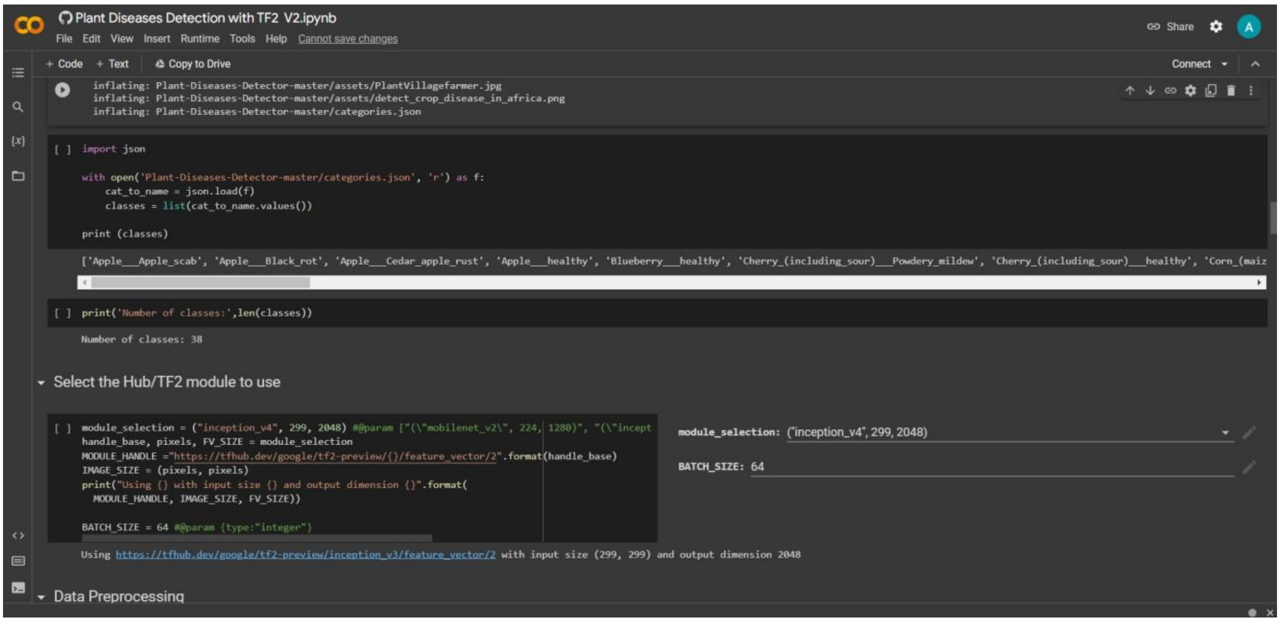


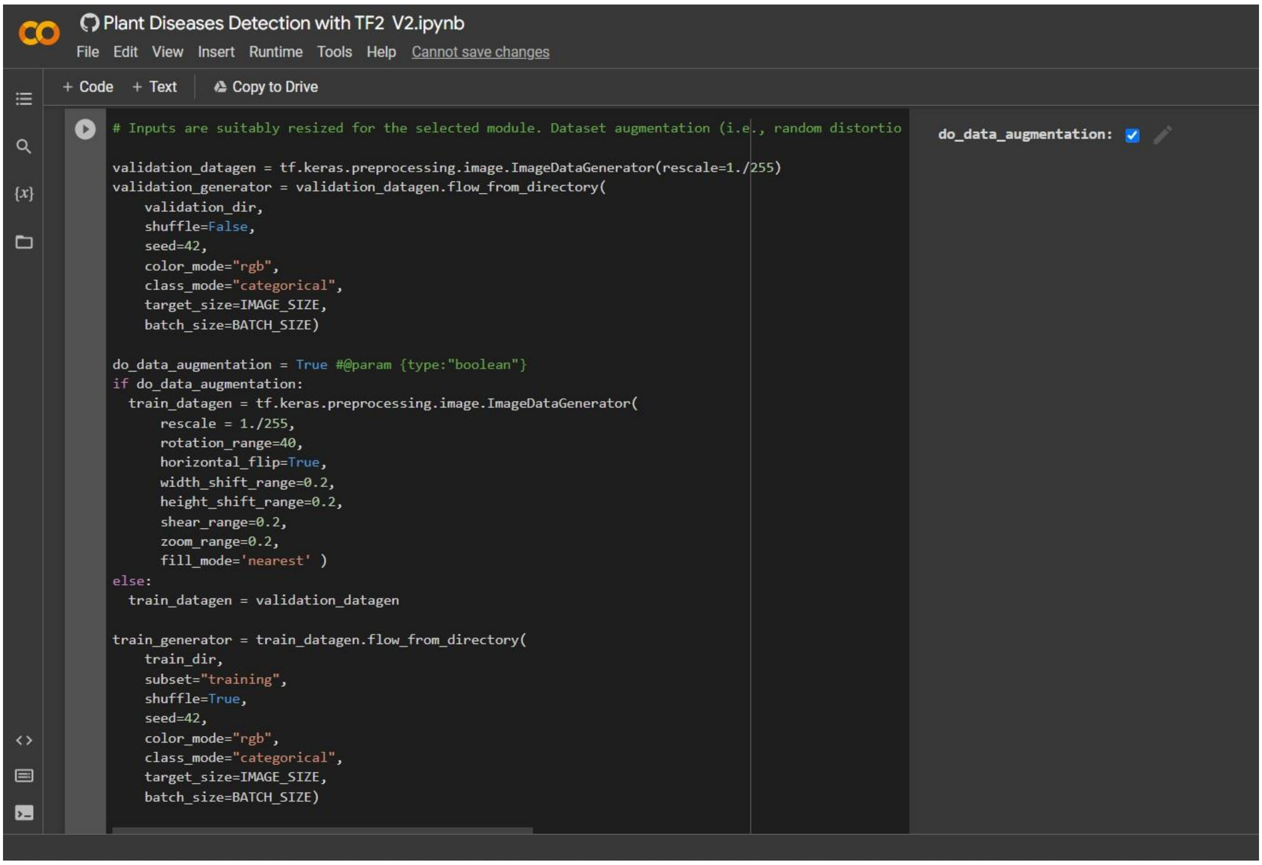
+ Code + Text Copy to Drive

Random sample images from validation dataset and predict

```
# Import OpenCV
import cv2

# Utility
import itertools
import random
from collections import Counter
from glob import iglob


def load_image(filename):
    img = cv2.imread(os.path.join(data_dir, validation_dir, filename))
    img = cv2.resize(img, (IMAGE_SIZE[0], IMAGE_SIZE[1]) )
    img = img /255

    return img


def predict(image):
    probabilities = model.predict(np.asarray([img]))[0]
    class_idx = np.argmax(probabilities)

    return {classes[class_idx]: probabilities[class_idx]}
```

```
for idx, filename in enumerate(random.sample(validation_generator.filenames, 5)):
    print("SOURCE: class: %s, file: %s" % (os.path.split(filename)[0], filename))

    img = load_image(filename)
    prediction = predict(img)
    print("PREDICTED: class: %s, confidence: %f" % (list(prediction.keys())[0], list(prediction.values())[0]))
    plt.imshow(img)
    plt.figure(idx)
    plt.show()
```


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
Share
Connect
inflating: Plant-Diseases-Detector-master/assets/PlantVillagefarmer.jpg
inflating: Plant-Diseases-Detector-master/assets/detect_crop_disease_in_africa.png
inflating: Plant-Diseases-Detector-master/categories.json
import json
with open('Plant-Diseases-Detector-master/categories.json', 'r') as f:
cat_to_name = json.load(f)
classes = list(cat_to_name.values())
print (classes)
['Apple___Apple_scab', 'Apple___Black_rot', 'Apple___Cedar_apple_rust', 'Apple___healthy', 'Blueberry___healthy', 'Cherry_(including_sour)___Powdery_mildew', 'Cherry_(including_sour)___healthy', 'Corn_(maiz
print('Number of classes:',len(classes))
Number of classes: 38
Select the Hub/TF2 module to use
module_selection = ("inception_v4", 299, 2048) #@param ["(\"mobilenet_v2\", 224, 1280)", "(\"incept
handle_base, pixels, FV_SIZE = module_selection
MODULE_HANDLE ="https://tfhub.dev/google/tf2-preview/{}/feature_vector/2".format(handle_base)
IMAGE_SIZE = (pixels, pixels)
print("Using {} with input size {} and output dimension {}".format(
MODULE_HANDLE, IMAGE_SIZE, FV_SIZE))
BATCH_SIZE = 64 #@param {type:"integer"}
module_selection: ("inception_v4", 299, 2048)
BATCH_SIZE: 64
Using https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/2 with input size (299, 299) and output dimension 2048
Data Preprocessing


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
# Inputs are suitably resized for the selected module. Dataset augmentation (i.e., random distortio
validation_datagen = tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
validation_generator = validation_datagen.flow_from_directory(
validation_dir,
shuffle=False,
seed=42,
color_mode="rgb",
class_mode="categorical",
target_size=IMAGE_SIZE,
batch_size=BATCH_SIZE)
do_data_augmentation = True #@param {type:"boolean"}
if do_data_augmentation:
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
rescale = 1./255,
rotation_range=40,
horizontal_flip=True,
width_shift_range=0.2,
height_shift_range=0.2,
shear_range=0.2,
zoom_range=0.2,
fill_mode='nearest')
else:
train_datagen = validation_datagen
train_generator = train_datagen.flow_from_directory(
train_dir,
subset="training",
shuffle=True,
seed=42,
color_mode="rgb",
class_mode="categorical",
target_size=IMAGE_SIZE,
batch_size=BATCH_SIZE)
do_data_augmentation:

Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

```
    shuffle=True,
    seed=42,
    color_mode="rgb",
    class_mode="categorical",
    target_size=IMAGE_SIZE,
    batch_size=BATCH_SIZE)
```

```
Found 10861 images belonging to 38 classes.
Found 43444 images belonging to 38 classes.
```

Build the model

All it takes is to put a linear classifier on top of the feature_extractor_layer with the Hub module.

For speed, we start out with a non-trainable feature_extractor_layer, but you can also enable fine-tuning for greater accuracy.

```
[ ] feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                       input_shape=IMAGE_SIZE+(3,),
                                       output_shape=[FV_SIZE])
```

```
[ ] do_fine_tuning = False #@param {type:"boolean"}
    if do_fine_tuning:
      feature_extractor.trainable = True
      # unfreeze some layers of base network for fine-tuning
      for layer in base_model.layers[-30:]:
        layer.trainable =True

    else:
      feature_extractor.trainable = False
```

do_fine_tuning:

Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

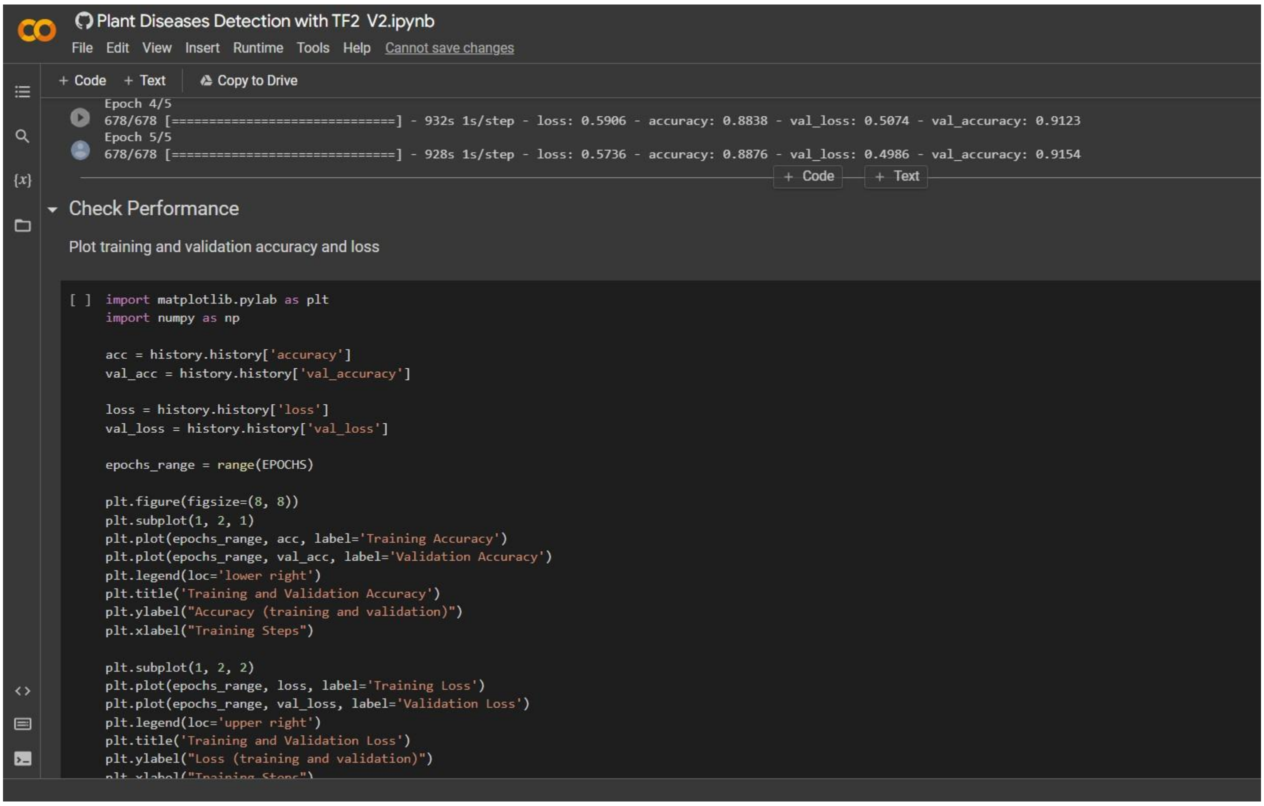
```
[ ] feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                       input_shape=IMAGE_SIZE+(3,),
                                       output_shape=[FV_SIZE])
```

```
[ ] do_fine_tuning = False #@param {type:"boolean"}
    if do_fine_tuning:
      feature_extractor.trainable = True
      # unfreeze some layers of base network for fine-tuning
      for layer in base_model.layers[-30:]:
        layer.trainable =True

    else:
      feature_extractor.trainable = False
```

do_fine_tuning:

```
[ ] print("Building model with", MODULE_HANDLE)
    model = tf.keras.Sequential([
        feature_extractor,
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dropout(rate=0.2),
        tf.keras.layers.Dense(train_generator.num_classes, activation='softmax',
                              kernel_regularizer=tf.keras.regularizers.l2(0.0001))
    ])
    #model.build((None,)+IMAGE_SIZE+(3,))

    model.summary()
```

```
Building model with https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/2
Model: "sequential"

Layer (type)                 Output Shape              Param #
```



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

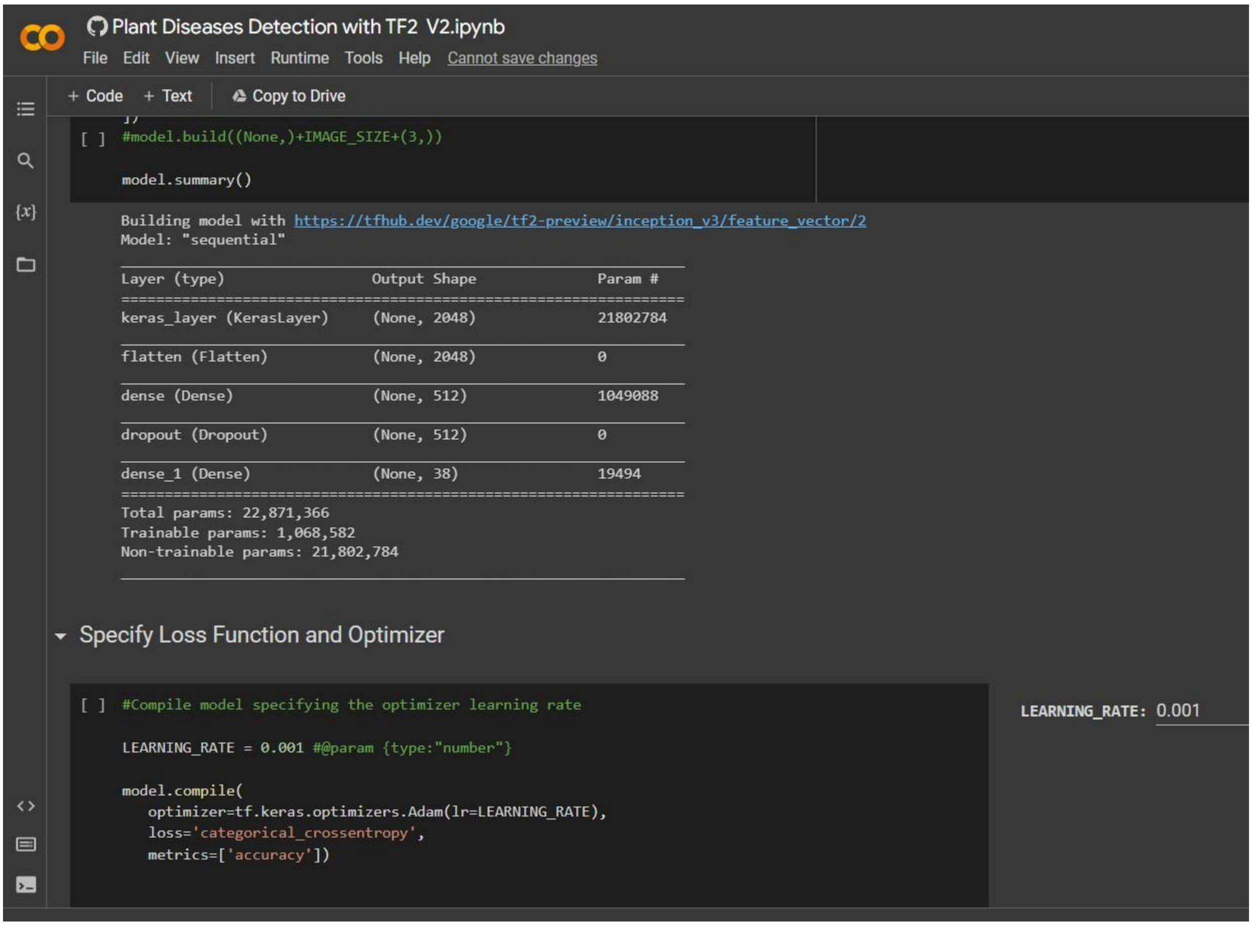
+ Code + Text Copy to Drive

```
Epoch 4/5
678/678 [==============================] - 932s 1s/step - loss: 0.5906 - accuracy: 0.8838 - val_loss: 0.5074 - val_accuracy: 0.9123
Epoch 5/5
678/678 [==============================] - 928s 1s/step - loss: 0.5736 - accuracy: 0.8876 - val_loss: 0.4986 - val_accuracy: 0.9154
```

Check Performance

Plot training and validation accuracy and loss

```
[ ] import matplotlib.pylab as plt
    import numpy as np

    acc = history.history['accuracy']
    val_acc = history.history['val_accuracy']

    loss = history.history['loss']
    val_loss = history.history['val_loss']

    epochs_range = range(EPOCHS)

    plt.figure(figsize=(8, 8))
    plt.subplot(1, 2, 1)
    plt.plot(epochs_range, acc, label='Training Accuracy')
    plt.plot(epochs_range, val_acc, label='Validation Accuracy')
    plt.legend(loc='lower right')
    plt.title('Training and Validation Accuracy')
    plt.ylabel("Accuracy (training and validation)")
    plt.xlabel("Training Steps")

    plt.subplot(1, 2, 2)
    plt.plot(epochs_range, loss, label='Training Loss')
    plt.plot(epochs_range, val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.title('Training and Validation Loss')
    plt.ylabel("Loss (training and validation)")
    plt.xlabel("Training Steps")
```



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

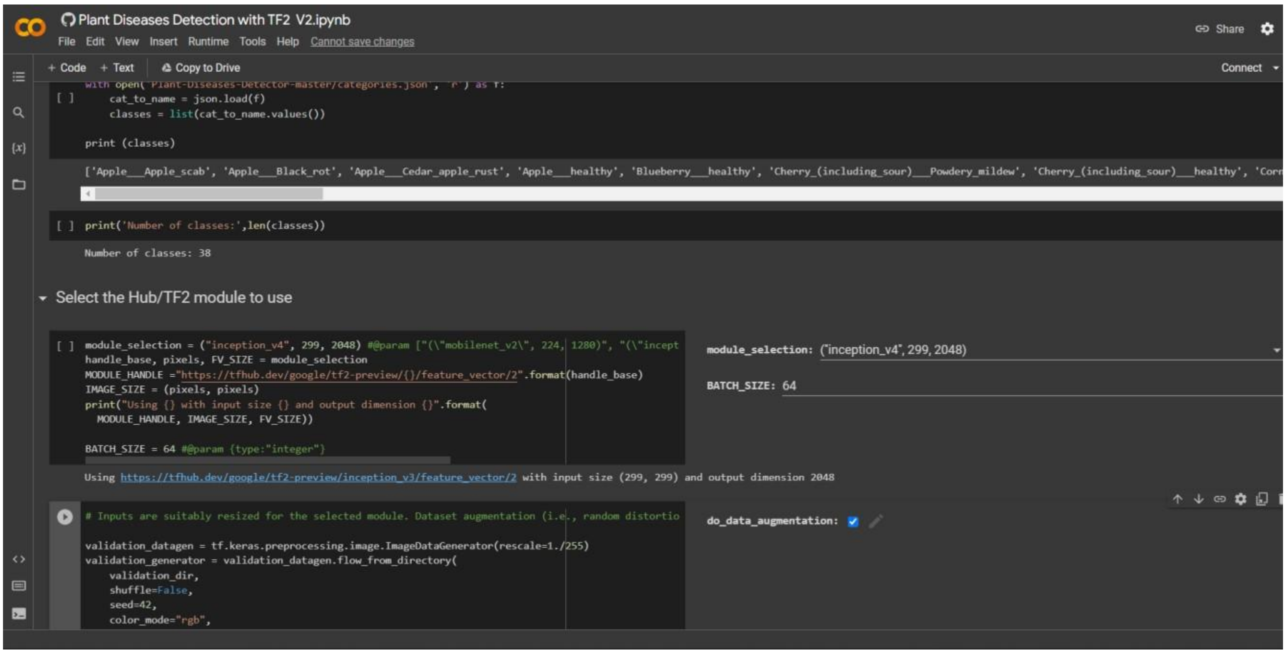
+ Code + Text Copy to Drive

```
[ ] #model.build((None,)+IMAGE_SIZE+(3,))

    model.summary()

Building model with https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/2
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
keras_layer (KerasLayer)     (None, 2048)              21802784
_________________________________________________________________
flatten (Flatten)            (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 512)               1049088
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 38)                19494
=================================================================
Total params: 22,871,366
Trainable params: 1,068,582
Non-trainable params: 21,802,784
_________________________________________________________________
```

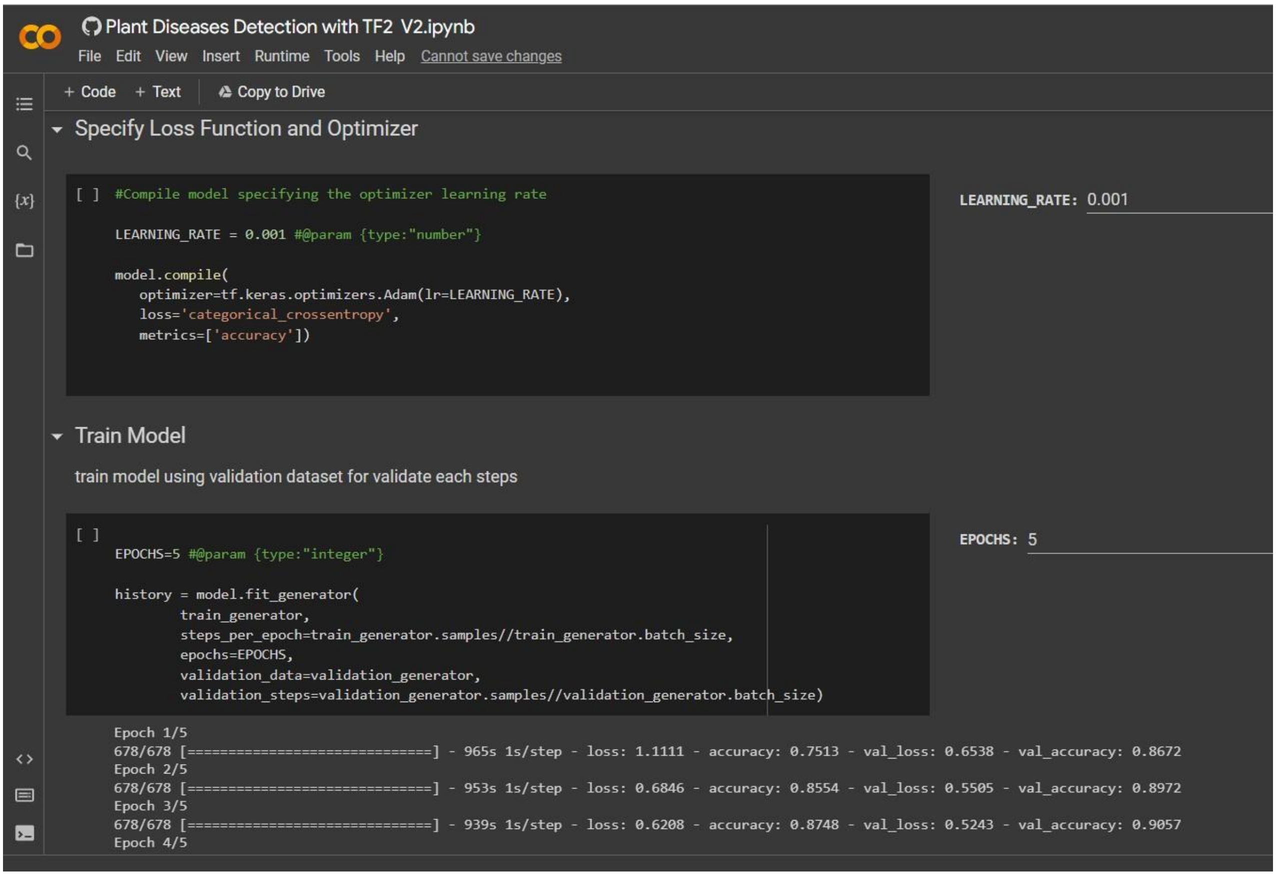
Specify Loss Function and Optimizer

```
[ ] #Compile model specifying the optimizer learning rate

    LEARNING_RATE = 0.001 #@param {type:"number"}

    model.compile(
       optimizer=tf.keras.optimizers.Adam(lr=LEARNING_RATE),
       loss='categorical_crossentropy',
       metrics=['accuracy'])
```

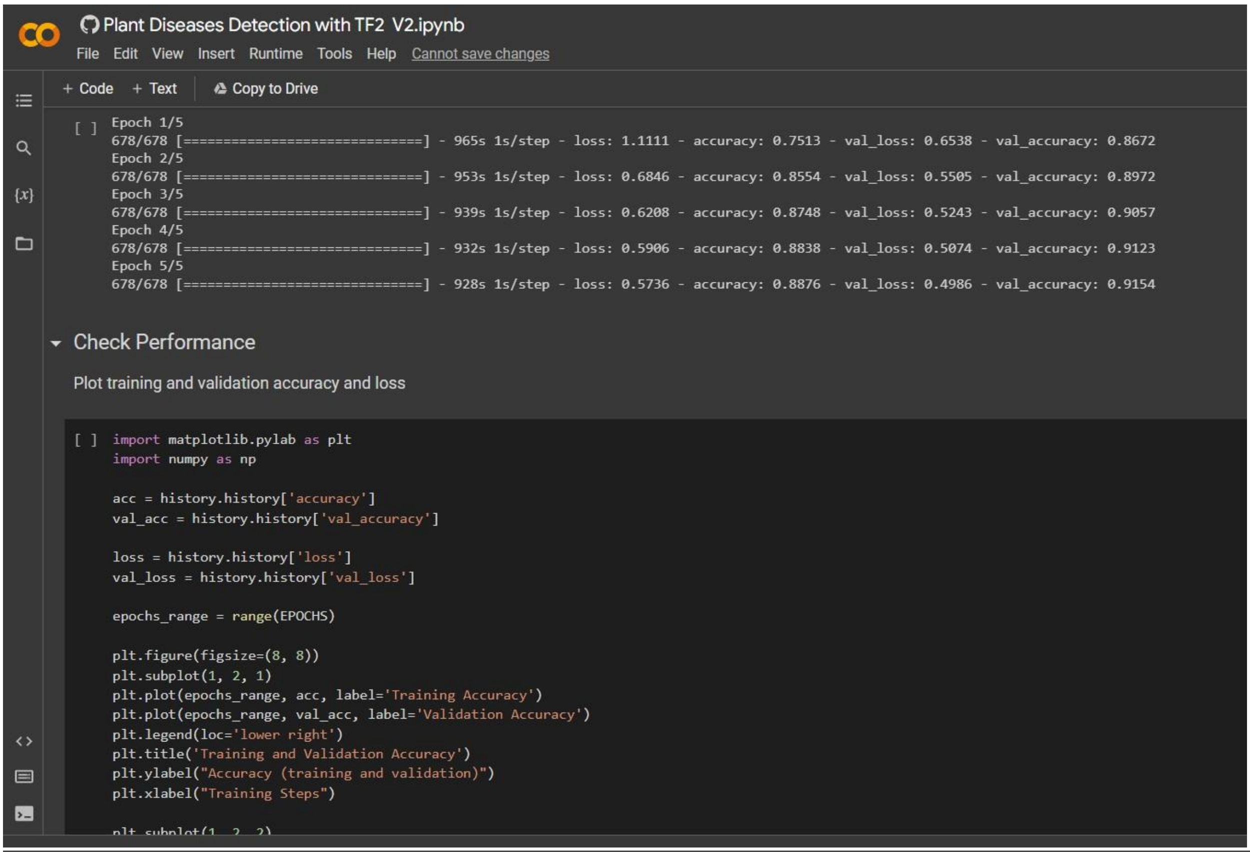
LEARNING_RATE: 0.001



Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
Share
+ Code + Text Copy to Drive
Connect
with open('Plant-Diseases-Detector-master/categories.json', 'r') as f:
cat_to_name = json.load(f)
classes = list(cat_to_name.values())
print (classes)
['Apple___Apple_scab', 'Apple___Black_rot', 'Apple___Cedar_apple_rust', 'Apple___healthy', 'Blueberry___healthy', 'Cherry_(including_sour)___Powdery_mildew', 'Cherry_(including_sour)___healthy', 'Corn
print('Number of classes:',len(classes))
Number of classes: 38
Select the Hub/TF2 module to use
module_selection = ("inception_v4", 299, 2048) #@param ["(\"mobilenet_v2\", 224, 1280)", "(\"incept
handle_base, pixels, FV_SIZE = module_selection
MODULE_HANDLE ="https://tfhub.dev/google/tf2-preview/{}/feature_vector/2".format(handle_base)
IMAGE_SIZE = (pixels, pixels)
print("Using {} with input size {} and output dimension {}".format(
MODULE_HANDLE, IMAGE_SIZE, FV_SIZE))
BATCH_SIZE = 64 #@param {type:"integer"}
module_selection: ("inception_v4", 299, 2048)
BATCH_SIZE: 64
Using https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/2 with input size (299, 299) and output dimension 2048
# Inputs are suitably resized for the selected module. Dataset augmentation (i.e., random distortio
validation_datagen = tf.keras.preprocessing.image.ImageDataGenerator(rescale=1./255)
validation_generator = validation_datagen.flow_from_directory(
validation_dir,
shuffle=False,
seed=42,
color_mode="rgb",
do_data_augmentation:



Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
Specify Loss Function and Optimizer
#Compile model specifying the optimizer learning rate
LEARNING_RATE = 0.001 #@param {type:"number"}
model.compile(
optimizer=tf.keras.optimizers.Adam(lr=LEARNING_RATE),
loss='categorical_crossentropy',
metrics=['accuracy'])
LEARNING_RATE: 0.001
Train Model
train model using validation dataset for validate each steps
EPOCHS=5 #@param {type:"integer"}
history = model.fit_generator(
train_generator,
steps_per_epoch=train_generator.samples//train_generator.batch_size,
epochs=EPOCHS,
validation_data=validation_generator,
validation_steps=validation_generator.samples//validation_generator.batch_size)
EPOCHS: 5
Epoch 1/5
678/678 [==============================] - 965s 1s/step - loss: 1.1111 - accuracy: 0.7513 - val_loss: 0.6538 - val_accuracy: 0.8672
Epoch 2/5
678/678 [==============================] - 953s 1s/step - loss: 0.6846 - accuracy: 0.8554 - val_loss: 0.5505 - val_accuracy: 0.8972
Epoch 3/5
678/678 [==============================] - 939s 1s/step - loss: 0.6208 - accuracy: 0.8748 - val_loss: 0.5243 - val_accuracy: 0.9057
Epoch 4/5



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

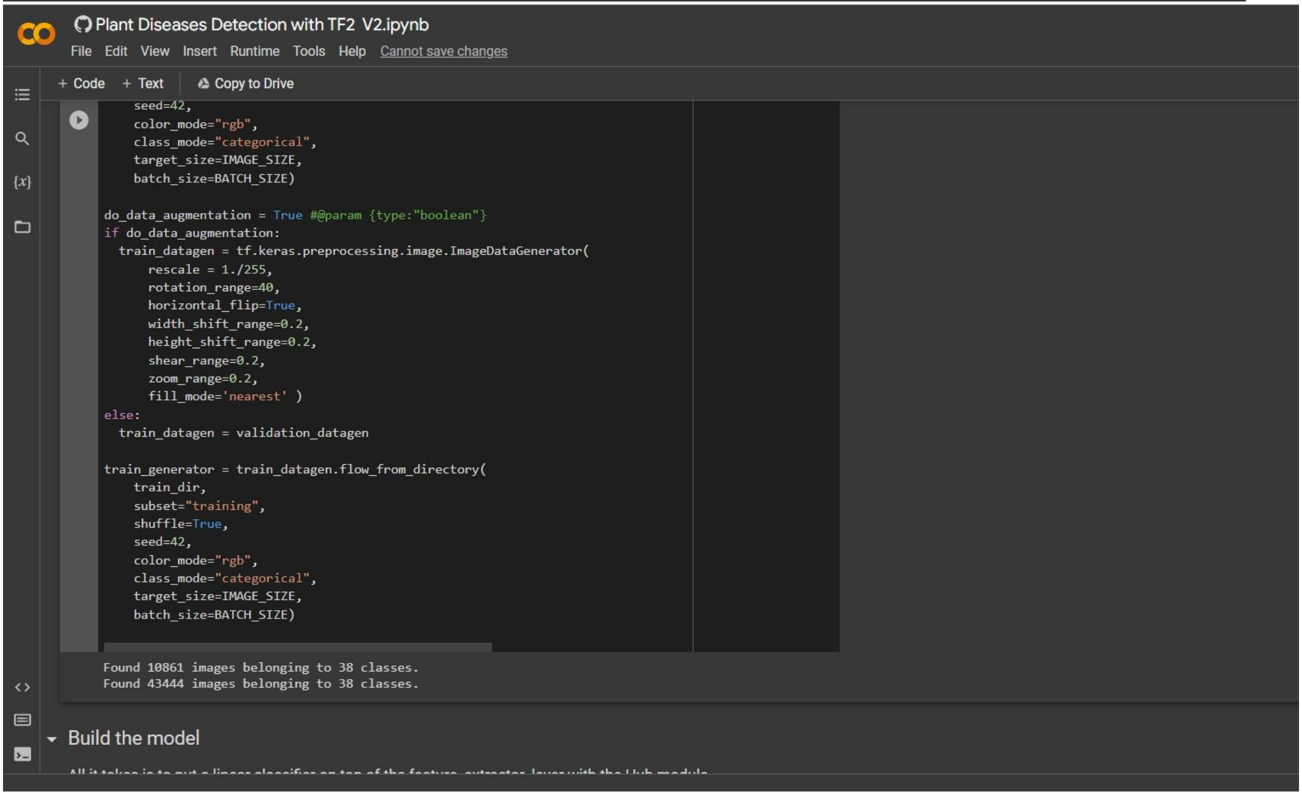
+ Code + Text Copy to Drive

```
Epoch 1/5
678/678 [==============================] - 965s 1s/step - loss: 1.1111 - accuracy: 0.7513 - val_loss: 0.6538 - val_accuracy: 0.8672
Epoch 2/5
678/678 [==============================] - 953s 1s/step - loss: 0.6846 - accuracy: 0.8554 - val_loss: 0.5505 - val_accuracy: 0.8972
Epoch 3/5
678/678 [==============================] - 939s 1s/step - loss: 0.6208 - accuracy: 0.8748 - val_loss: 0.5243 - val_accuracy: 0.9057
Epoch 4/5
678/678 [==============================] - 932s 1s/step - loss: 0.5906 - accuracy: 0.8838 - val_loss: 0.5074 - val_accuracy: 0.9123
Epoch 5/5
678/678 [==============================] - 928s 1s/step - loss: 0.5736 - accuracy: 0.8876 - val_loss: 0.4986 - val_accuracy: 0.9154
```

Check Performance

Plot training and validation accuracy and loss

```
import matplotlib.pylab as plt
import numpy as np

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(EPOCHS)

plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')
plt.ylabel("Accuracy (training and validation)")
plt.xlabel("Training Steps")

plt.subplot(1, 2, 2)
```



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

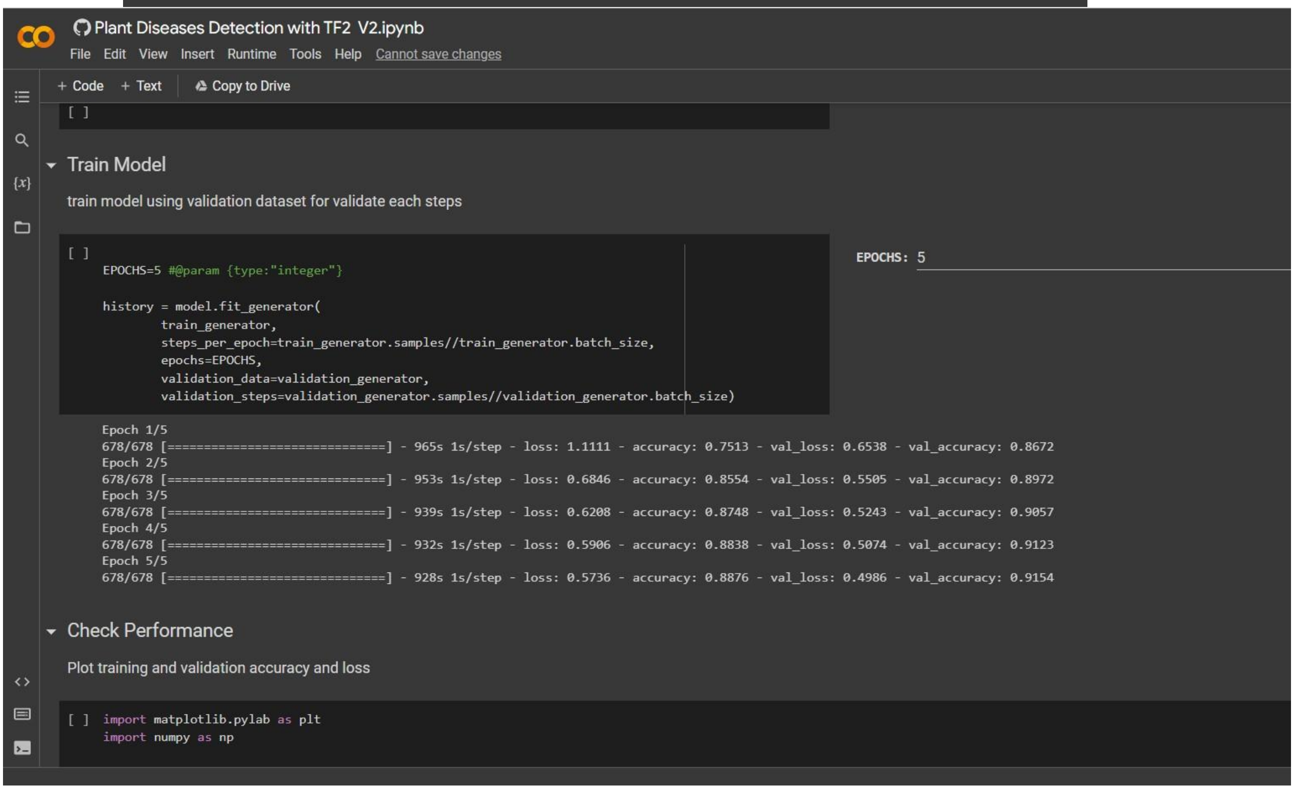
+ Code + Text Copy to Drive

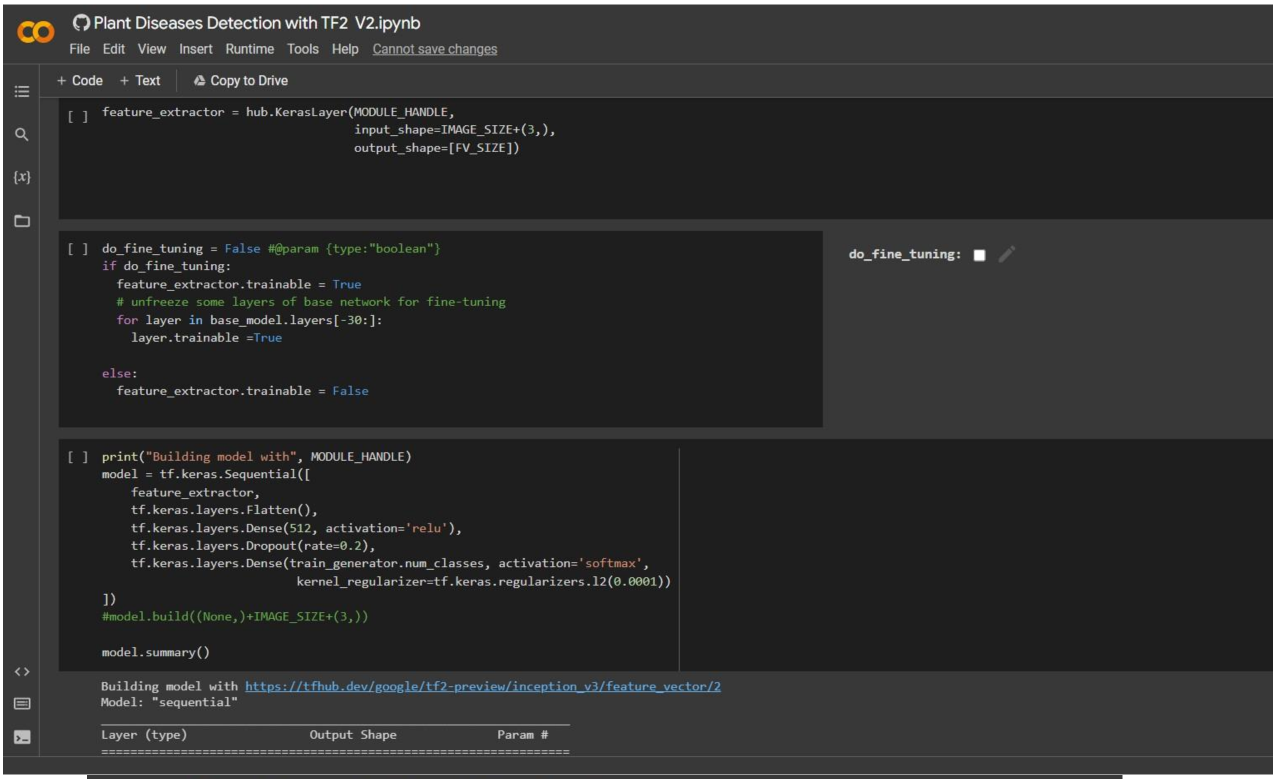
```
    seed=42,
    color_mode="rgb",
    class_mode="categorical",
    target_size=IMAGE_SIZE,
    batch_size=BATCH_SIZE)

do_data_augmentation = True #@param {type:"boolean"}
if do_data_augmentation:
  train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
      rescale = 1./255,
      rotation_range=40,
      horizontal_flip=True,
      width_shift_range=0.2,
      height_shift_range=0.2,
      shear_range=0.2,
      zoom_range=0.2,
      fill_mode='nearest' )
else:
  train_datagen = validation_datagen

train_generator = train_datagen.flow_from_directory(
    train_dir,
    subset="training",
    shuffle=True,
    seed=42,
    color_mode="rgb",
    class_mode="categorical",
    target_size=IMAGE_SIZE,
    batch_size=BATCH_SIZE)
```

```
Found 10861 images belonging to 38 classes.
Found 43444 images belonging to 38 classes.
```

Build the model


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
[ ]
plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.ylabel("Loss (training and validation)")
plt.xlabel("Training Steps")
plt.show()
Training and Validation Accuracy
Accuracy (training and validation)
Training Accuracy
Validation Accuracy
Training Steps
Training and Validation Loss
Loss (training and validation)
Training Loss
Validation Loss
Training Steps


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
[ ]
Train Model
train model using validation dataset for validate each steps
[ ]
EPOCHS=5 #@param {type:"integer"}
history = model.fit_generator(
train_generator,
steps_per_epoch=train_generator.samples//train_generator.batch_size,
epochs=EPOCHS,
validation_data=validation_generator,
validation_steps=validation_generator.samples//validation_generator.batch_size)
EPOCHS: 5
Epoch 1/5
678/678 [==============================] - 965s 1s/step - loss: 1.1111 - accuracy: 0.7513 - val_loss: 0.6538 - val_accuracy: 0.8672
Epoch 2/5
678/678 [==============================] - 953s 1s/step - loss: 0.6846 - accuracy: 0.8554 - val_loss: 0.5505 - val_accuracy: 0.8972
Epoch 3/5
678/678 [==============================] - 939s 1s/step - loss: 0.6208 - accuracy: 0.8748 - val_loss: 0.5243 - val_accuracy: 0.9057
Epoch 4/5
678/678 [==============================] - 932s 1s/step - loss: 0.5906 - accuracy: 0.8838 - val_loss: 0.5074 - val_accuracy: 0.9123
Epoch 5/5
678/678 [==============================] - 928s 1s/step - loss: 0.5736 - accuracy: 0.8876 - val_loss: 0.4986 - val_accuracy: 0.9154
Check Performance
Plot training and validation accuracy and loss
[ ] import matplotlib.pylab as plt
import numpy as np



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

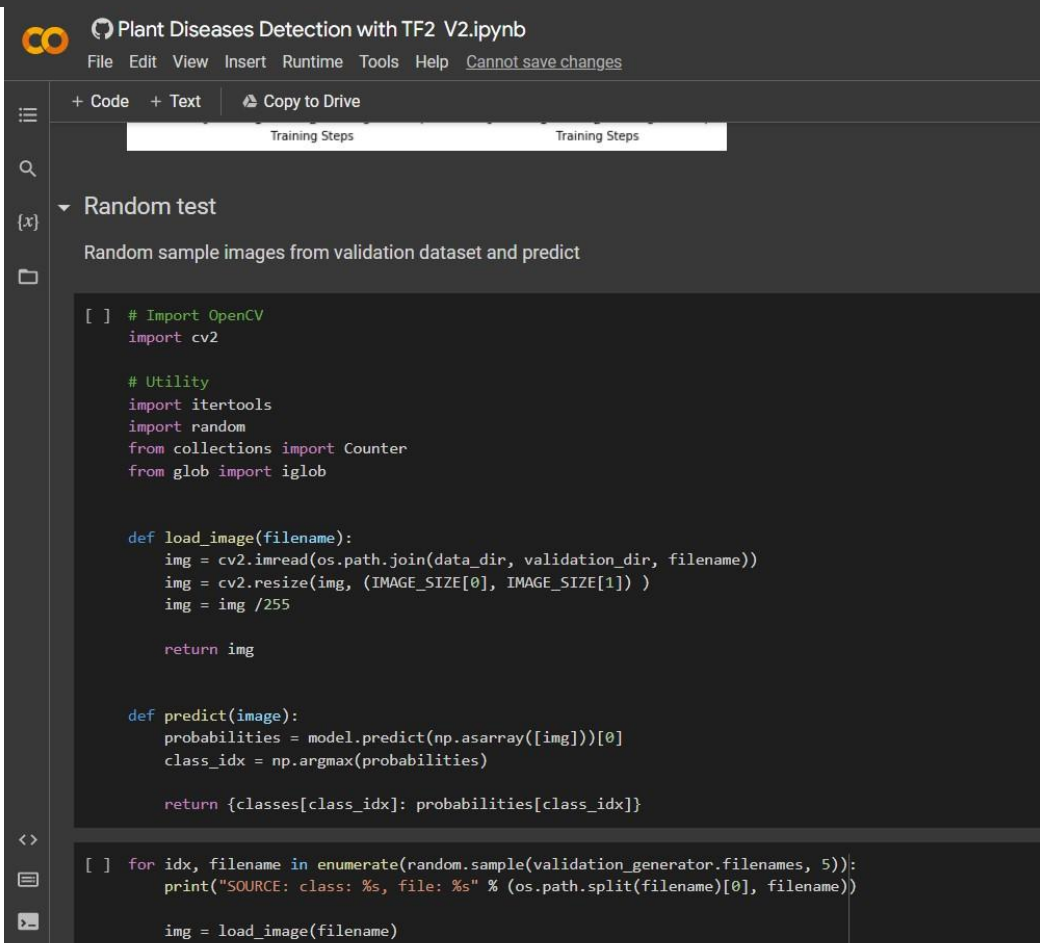
```
[ ] feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                        input_shape=IMAGE_SIZE+(3,),
                                        output_shape=[FV_SIZE])
```

```
[ ] do_fine_tuning = False #@param {type:"boolean"}
    if do_fine_tuning:
      feature_extractor.trainable = True
      # unfreeze some layers of base network for fine-tuning
      for layer in base_model.layers[-30:]:
        layer.trainable =True

    else:
      feature_extractor.trainable = False
```

do_fine_tuning:

```
[ ] print("Building model with", MODULE_HANDLE)
    model = tf.keras.Sequential([
        feature_extractor,
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dropout(rate=0.2),
        tf.keras.layers.Dense(train_generator.num_classes, activation='softmax',
                              kernel_regularizer=tf.keras.regularizers.l2(0.0001))
    ])
    #model.build((None,)+IMAGE_SIZE+(3,))

    model.summary()
```

```
Building model with https://tfhub.dev/google/tf2-preview/inception_v3/feature_vector/2
Model: "sequential"

Layer (type)                 Output Shape              Param #
=================================================================
```



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

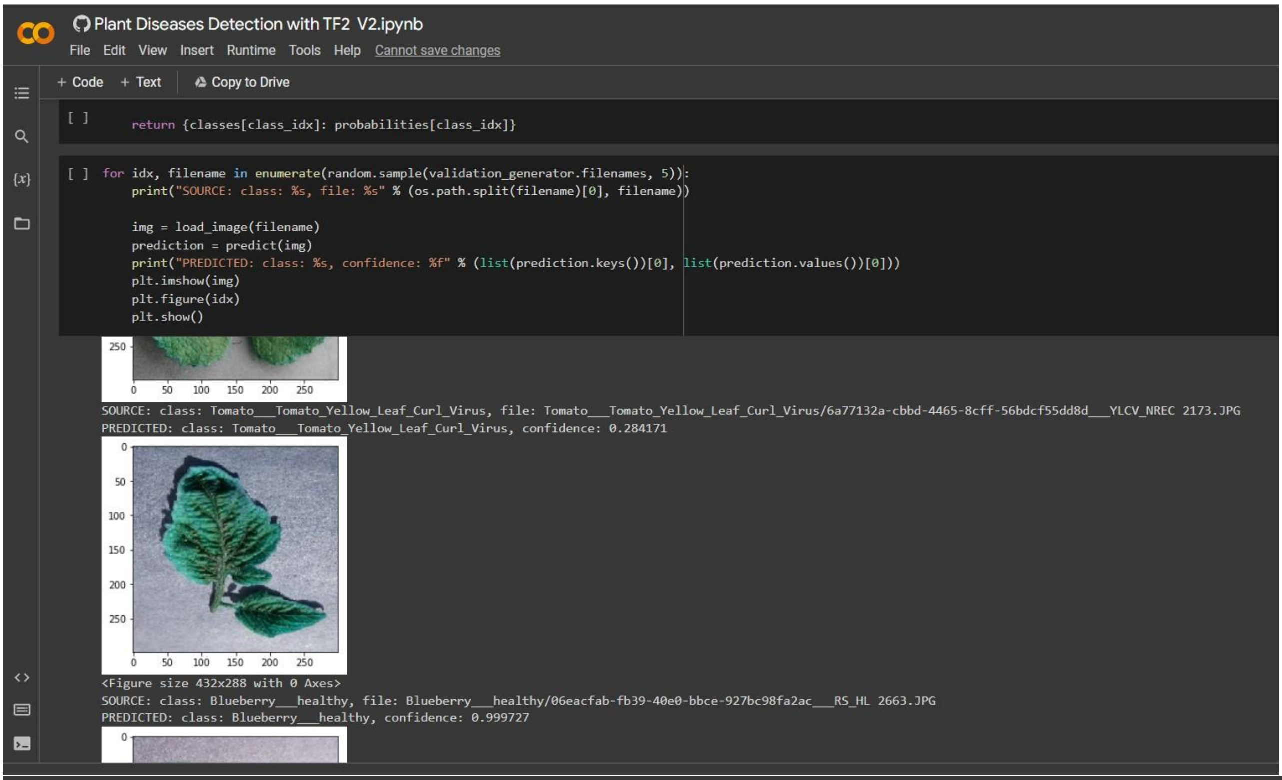
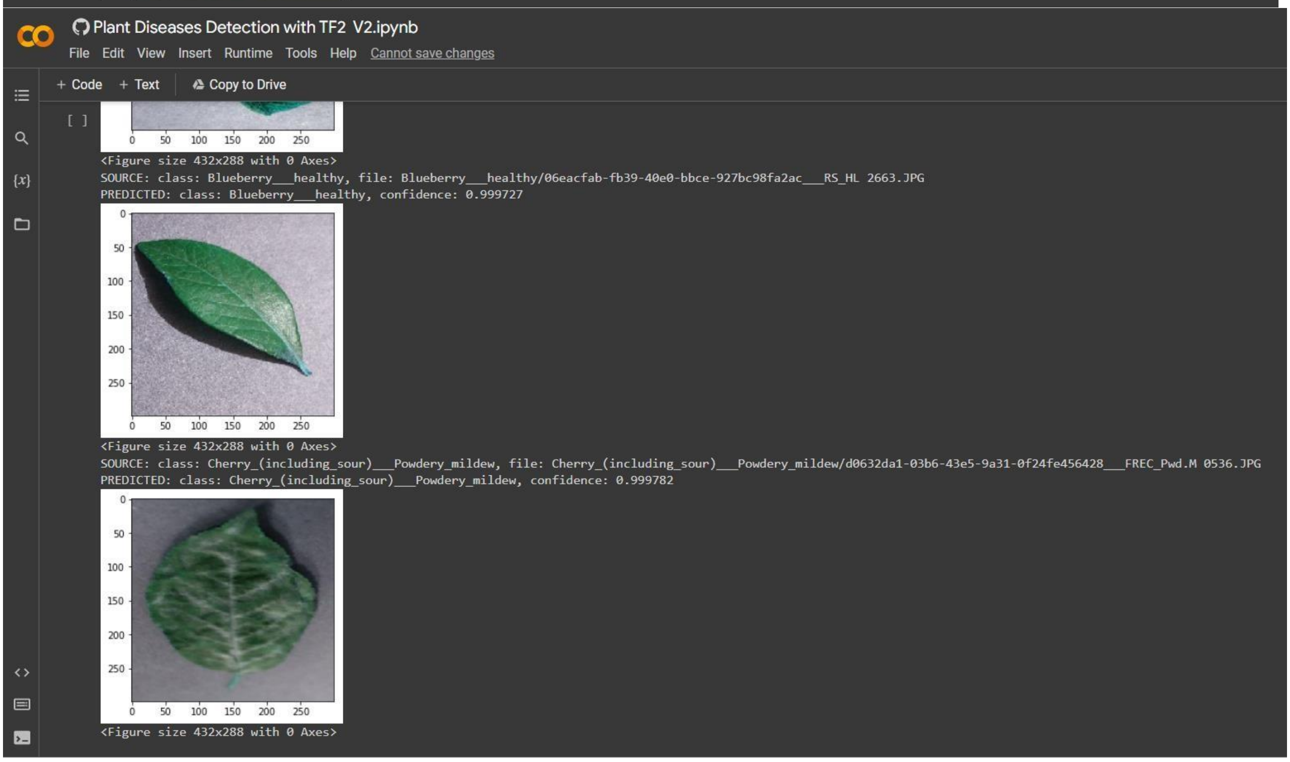
Training Steps Training Steps

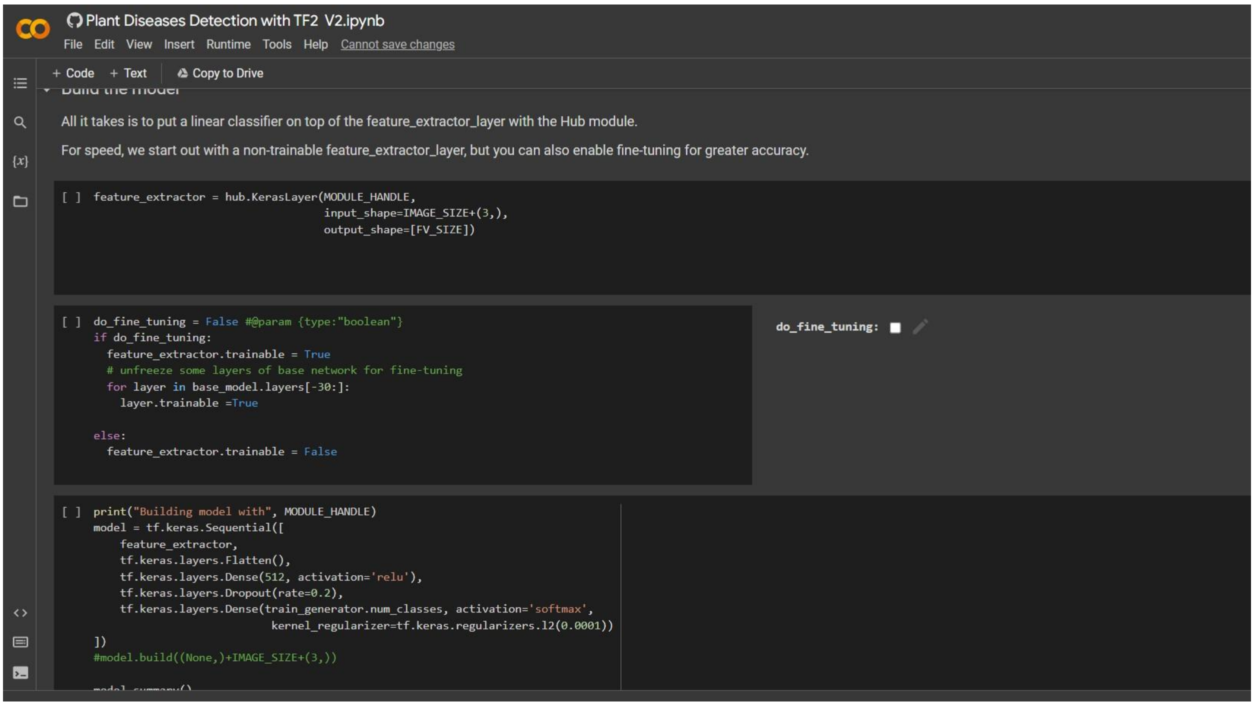
Random test

Random sample images from validation dataset and predict

```
[ ] # Import OpenCV
    import cv2

    # Utility
    import itertools
    import random
    from collections import Counter
    from glob import iglob


    def load_image(filename):
        img = cv2.imread(os.path.join(data_dir, validation_dir, filename))
        img = cv2.resize(img, (IMAGE_SIZE[0], IMAGE_SIZE[1]) )
        img = img /255

        return img


    def predict(image):
        probabilities = model.predict(np.asarray([img]))[0]
        class_idx = np.argmax(probabilities)

        return {classes[class_idx]: probabilities[class_idx]}
```

```
[ ] for idx, filename in enumerate(random.sample(validation_generator.filenames, 5)):
        print("SOURCE: class: %s, file: %s" % (os.path.split(filename)[0], filename))

        img = load_image(filename)
```


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
return {classes[class_idx]: probabilities[class_idx]}
for idx, filename in enumerate(random.sample(validation_generator.filenames, 5)):
print("SOURCE: class: %s, file: %s" % (os.path.split(filename)[0], filename))
img = load_image(filename)
prediction = predict(img)
print("PREDICTED: class: %s, confidence: %f" % (list(prediction.keys())[0], list(prediction.values())[0]))
plt.imshow(img)
plt.figure(idx)
plt.show()
SOURCE: class: Tomato___Tomato_Yellow_Leaf_Curl_Virus, file: Tomato___Tomato_Yellow_Leaf_Curl_Virus/6a77132a-cbbd-4465-8cff-56bdcf55dd8d___YLCV_NREC 2173.JPG
PREDICTED: class: Tomato___Tomato_Yellow_Leaf_Curl_Virus, confidence: 0.284171
<Figure size 432x288 with 0 Axes>
SOURCE: class: Blueberry___healthy, file: Blueberry___healthy/06eacfab-fb39-40e0-bbce-927bc98fa2ac___RS_HL 2663.JPG
PREDICTED: class: Blueberry___healthy, confidence: 0.999727

Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
<Figure size 432x288 with 0 Axes>
SOURCE: class: Blueberry___healthy, file: Blueberry___healthy/06eacfab-fb39-40e0-bbce-927bc98fa2ac___RS_HL 2663.JPG
PREDICTED: class: Blueberry___healthy, confidence: 0.999727
<Figure size 432x288 with 0 Axes>
SOURCE: class: Cherry_(including_sour)___Powdery_mildew, file: Cherry_(including_sour)___Powdery_mildew/d0632da1-03b6-43e5-9a31-0f24fe456428___FREC_Pwd.M 0536.JPG
PREDICTED: class: Cherry_(including_sour)___Powdery_mildew, confidence: 0.999782
<Figure size 432x288 with 0 Axes>



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

Build the model

All it takes is to put a linear classifier on top of the feature_extractor_layer with the Hub module.

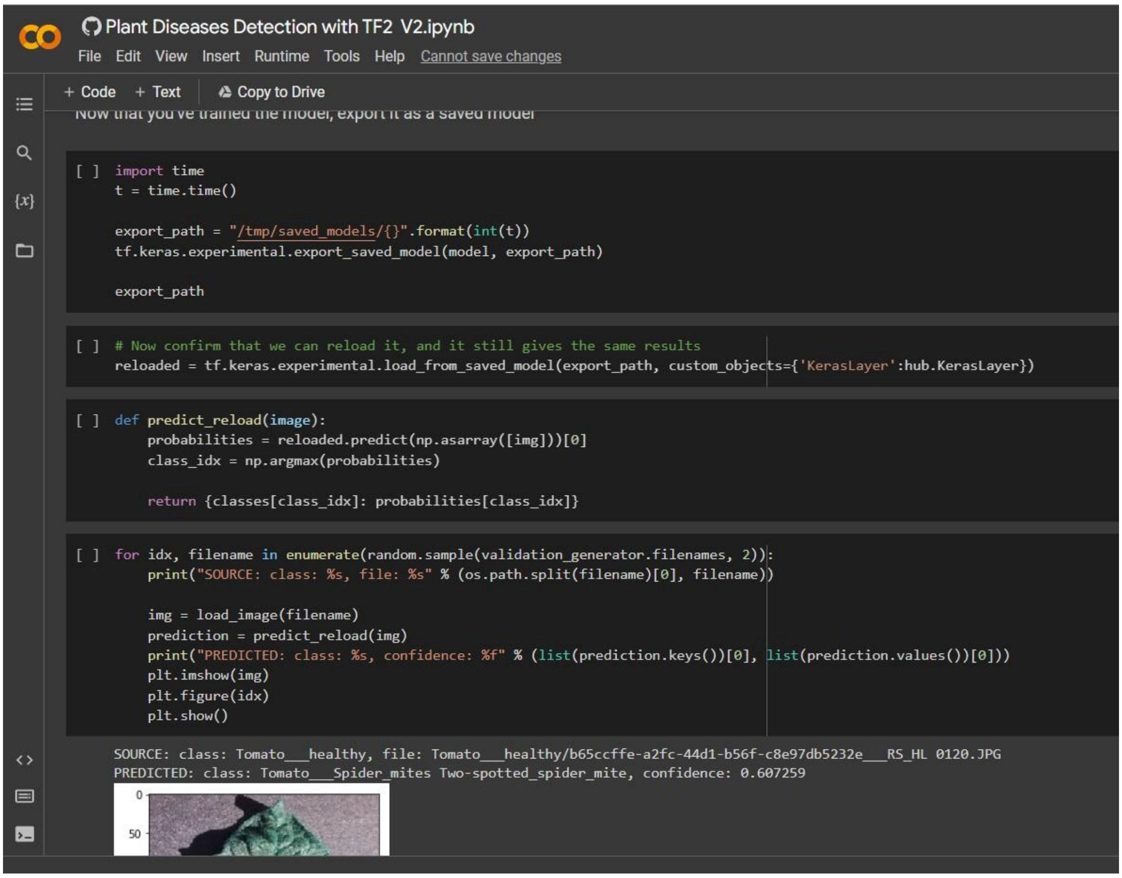
For speed, we start out with a non-trainable feature_extractor_layer, but you can also enable fine-tuning for greater accuracy.

```
[ ] feature_extractor = hub.KerasLayer(MODULE_HANDLE,
                                       input_shape=IMAGE_SIZE+(3,),
                                       output_shape=[FV_SIZE])
```

```
[ ] do_fine_tuning = False #@param {type:"boolean"}
    if do_fine_tuning:
      feature_extractor.trainable = True
      # unfreeze some layers of base network for fine-tuning
      for layer in base_model.layers[-30:]:
        layer.trainable =True

    else:
      feature_extractor.trainable = False
```

do_fine_tuning:

```
[ ] print("Building model with", MODULE_HANDLE)
    model = tf.keras.Sequential([
        feature_extractor,
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(512, activation='relu'),
        tf.keras.layers.Dropout(rate=0.2),
        tf.keras.layers.Dense(train_generator.num_classes, activation='softmax',
                              kernel_regularizer=tf.keras.regularizers.l2(0.0001))
    ])
    #model.build((None,)+IMAGE_SIZE+(3,))

    model.summary()
```



Plant Diseases Detection with TF2 V2.ipynb

File Edit View Insert Runtime Tools Help Cannot save changes

+ Code + Text Copy to Drive

Now that you've trained the model, export it as a saved model

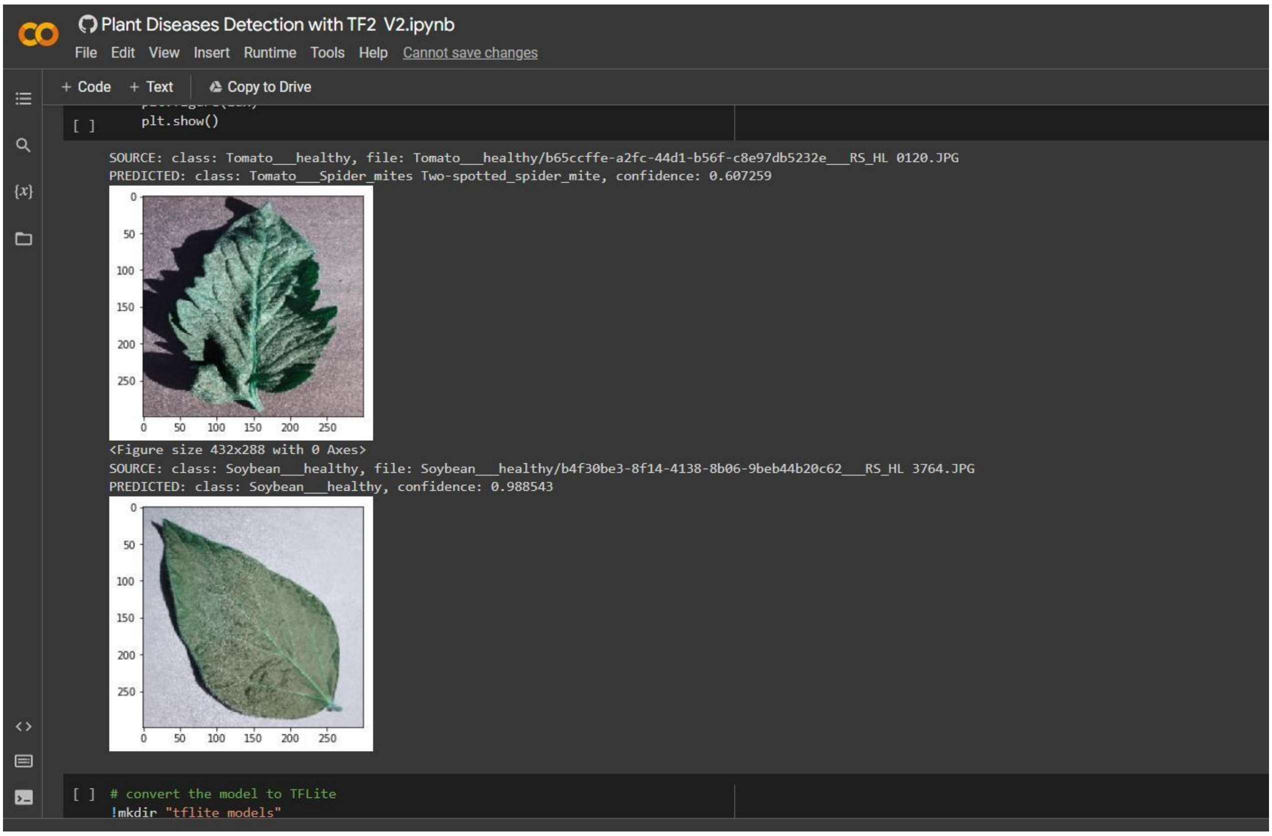
```
[ ] import time
    t = time.time()

    export_path = "/tmp/saved_models/{}".format(int(t))
    tf.keras.experimental.export_saved_model(model, export_path)

    export_path
```

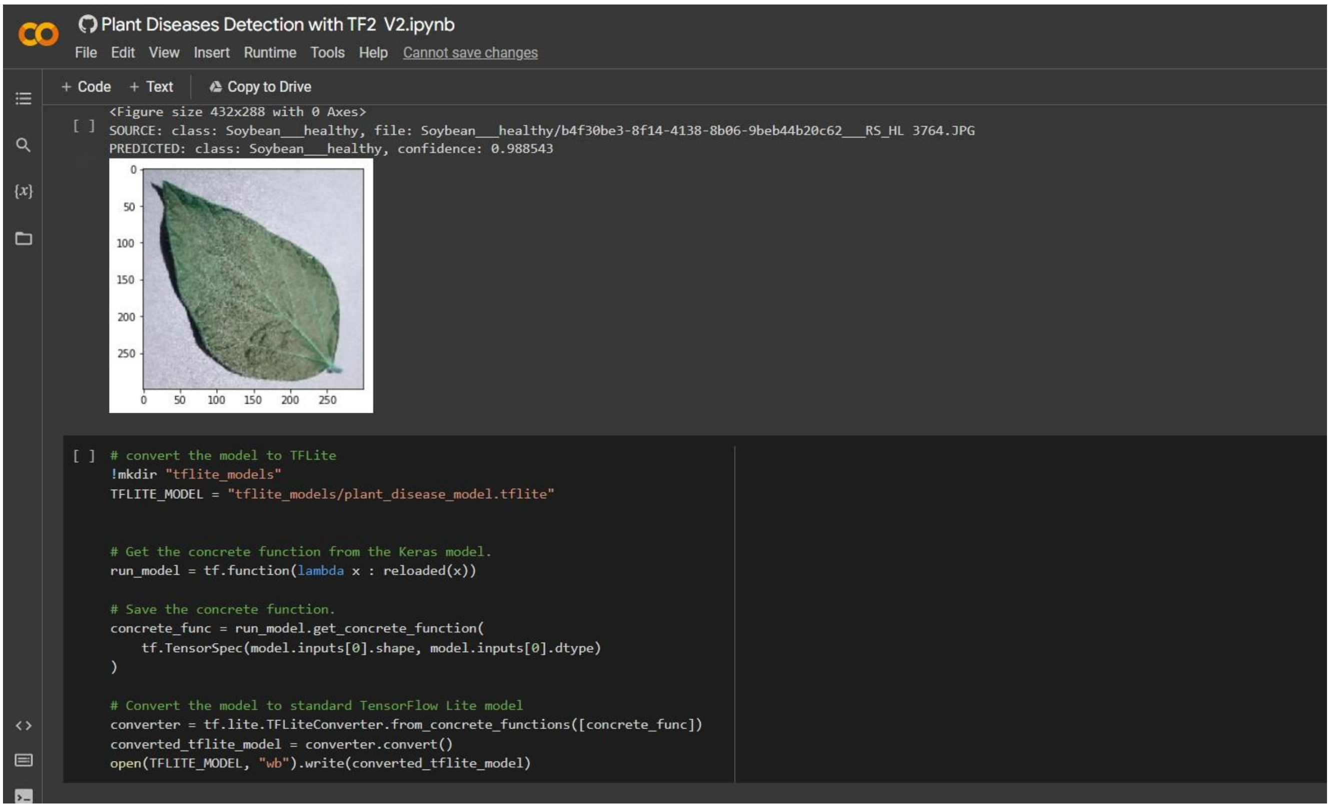
```
[ ] # Now confirm that we can reload it, and it still gives the same results
    reloaded = tf.keras.experimental.load_from_saved_model(export_path, custom_objects={'KerasLayer':hub.KerasLayer})
```

```
[ ] def predict_reload(image):
        probabilities = reloaded.predict(np.asarray([img]))[0]
        class_idx = np.argmax(probabilities)

        return {classes[class_idx]: probabilities[class_idx]}
```

```
[ ] for idx, filename in enumerate(random.sample(validation_generator.filenames, 2)):
        print("SOURCE: class: %s, file: %s" % (os.path.split(filename)[0], filename))

        img = load_image(filename)
        prediction = predict_reload(img)
        print("PREDICTED: class: %s, confidence: %f" % (list(prediction.keys())[0], list(prediction.values())[0]))
        plt.imshow(img)
        plt.figure(idx)
        plt.show()
```

```
SOURCE: class: Tomato___healthy, file: Tomato___healthy/b65ccffe-a2fc-44d1-b56f-c8e97db5232e___RS_HL 0120.JPG
PREDICTED: class: Tomato___Spider_mites Two-spotted_spider_mite, confidence: 0.607259
```


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
[ ] plt.show()
SOURCE: class: Tomato___healthy, file: Tomato___healthy/b65ccffe-a2fc-44d1-b56f-c8e97db5232e___RS_HL 0120.JPG
PREDICTED: class: Tomato___Spider_mites Two-spotted_spider_mite, confidence: 0.607259
<Figure size 432x288 with 0 Axes>
SOURCE: class: Soybean___healthy, file: Soybean___healthy/b4f30be3-8f14-4138-8b06-9beb44b20c62___RS_HL 3764.JPG
PREDICTED: class: Soybean___healthy, confidence: 0.988543
[ ] # convert the model to TFLite
!mkdir "tflite_models"


Plant Diseases Detection with TF2 V2.ipynb
File Edit View Insert Runtime Tools Help Cannot save changes
+ Code + Text Copy to Drive
<Figure size 432x288 with 0 Axes>
[ ] SOURCE: class: Soybean___healthy, file: Soybean___healthy/b4f30be3-8f14-4138-8b06-9beb44b20c62___RS_HL 3764.JPG
PREDICTED: class: Soybean___healthy, confidence: 0.988543
[ ] # convert the model to TFLite
!mkdir "tflite_models"
TFLITE_MODEL = "tflite_models/plant_disease_model.tflite"
# Get the concrete function from the Keras model.
run_model = tf.function(lambda x : reloaded(x))
# Save the concrete function.
concrete_func = run_model.get_concrete_function(
tf.TensorSpec(model.inputs[0].shape, model.inputs[0].dtype)
)
# Convert the model to standard TensorFlow Lite model
converter = tf.lite.TFLiteConverter.from_concrete_functions([concrete_func])
converted_tflite_model = converter.convert()
open(TFLITE_MODEL, "wb").write(converted_tflite_model)

PAPER PUBLICATION STATUS

Dr. P Kanmani, Ms. Anushka S, Ms. Anushka Agarwal- Crop Disease Prediction Using IOT And Deep Neural Networks. Submitted the paper in ICIOT-2023 conference.

PLAGIARISM REPORT

Crop disease prediction using IOT and deep neural networks

ORIGINALITY REPORT

7%	5%	3%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to College of Banking and Financial Studies Student Paper	2%
2	Submitted to SRM University Student Paper	1%
3	Submitted to Indian Institute of Technology Patna Student Paper	1%
4	Submitted to California Lutheran University Student Paper	<1%
5	Submitted to Coventry University Student Paper	<1%
6	Submitted to Cerritos College Student Paper	<1%
7	www.bartleby.com Internet Source	<1%
8	Amit Kumar Tyagi, Niladhuri Sreenath. "Intelligent Transportation Systems: Theory	<1%