# Project Name : Plant Disease Identifier

| | |
|---|---|
| Instructor : | **Prof. Graham Wall** |
| Students : | 1) Aditya Rai C0793398<br>2) Darshil Chikani C0803705<br>3) Kanishak Gautam C0785569<br>4) Tatiana Klimova C0775489<br>5) Kiranjot Jaswal C0780763<br>6) Navdeep Kaur C0791868 |
| Class : | **CSD-3444** |

**INDEX**

# CHAPTER 1: INTODUCTION ABOUT PROJECT

## 1.1 Overview of project

We depend on edible plants just as we depend on oxygen. Without crops, there is no food, and without food, there is no life. It's no accident that human civilization began to thrive with the invention of agriculture.

Today, modern technology allows us to grow crops in quantities necessary for a steady food supply for billions of people. But diseases remain a major threat to this supply, and a large fraction of crops are lost each year to diseases. The situation is particularly dire for the 500 million smallholder farmers around the globe, whose livelihoods depend on their crops doing well. In Africa alone, 80% of the agricultural output comes from smallholder farmers.

With billions of smartphones around the globe, wouldn't it be great if the smartphone could be turned into a disease diagnostics tool, recognizing diseases from images it captures with its camera? This challenge is the first of many steps turning this vision into a reality. PlantVillage is a not-for-profit project by Penn State University in the US and EPFL in Switzerland. We have collected - and continue to collect - tens of thousands of images of diseased and healthy crops. The goal of this challenge is to develop algorithms than can accurately diagnose a disease based on an image.

## 1.2 Scope of project

This project is able to solve mainly farmer’s problems. Farmers can use this application to identify the disease by this application. Botany students can also learn about plants and their disease.

## 1.3 Purpose of project

The purpose of this project is to help the farmers to identify the disease for plant which he is going seed in the farm and it is also helpful to botany

students to study about plants, and their leaves. And they can also study about the disease of varies plants.

# CHAPTER 2 : HARDWARE AND SOFTWARE REQUIREMENTS

## 2.1Developer Requirement

For developer, it requires minimum 8GB RAM, 10GB storage requirement, and software develop on Windows 7, Windows 8, Windows 10, MacOS or Linux. It requires python environment.

## 2.2 User Requirement

Software can run on Windows 7, Windows 8, and Windows 10 platform. It requires 1 GB RAM and 10GB storage space.

# CHAPTER 3 : PROJECT MODULES

## 3.1 Pooling Layer

It is common to periodically insert a Pooling layer in-between successive Conv layers in a ConvNet architecture. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. The Pooling Layer operates independently on every depth slice of the input and resizes it spatially, using the MAX operation. The most common form is a pooling layer with filters of size 2x2 applied with a stride of 2 downsamples every depth slice in the input by 2 along both width and height, discarding 75% of the activations. Every MAX operation would in this case be taking a max over 4 numbers (little 2x2 region in some depth slice). The depth dimension remains unchanged. More generally, the pooling layer:

Accepts a volume of size W1×H1×D1W1×H1×D1

Requires two hyperparameters:

their spatial extent FF,

the stride SS,

Produces a volume of size W2×H2×D2W2×H2×D2 where:

W2=(W1−F)/S+1W2=(W1−F)/S+1

H2=(H1−F)/S+1H2=(H1−F)/S+1

D2=D1D2=D1

Introduces zero parameters since it computes a fixed function of the input

For Pooling layers, it is not common to pad the input using zero-padding.

It is worth noting that there are only two commonly seen variations of the max pooling layer found in practice: A pooling layer with F=3,S=2F=3,S=2 (also called overlapping pooling), and more commonly F=2,S=2F=2,S=2. Pooling sizes with larger receptive fields are too destructive.

## 3.2 Transfer Learning

Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task.

Transfer learning is related to problems such as multi-task learning and concept drift and is not exclusively an area of study for deep learning.

Nevertheless, transfer learning is popular in deep learning given the enormous resources required to train deep learning models or the large and challenging datasets on which deep learning models are trained.

Transfer learning only works in deep learning if the model features learned from the first task are general.

## 3.3 ResNet-50

ResNet is a short name for Residual Network. As the name of the network indicates, the new terminology that this network introduces is residual learning.

Deep convolutional neural networks have led to a series of breakthroughs for image classification. Many other visual recognition tasks have also greatly benefited from very deep models. So, over the years there is a trend to go more deeper, to solve more complex tasks and to also increase /improve the classification/recognition accuracy. But, as we go deeper; the training of neural network becomes difficult and also the accuracy starts saturating and then degrades also. Residual Learning tries to solve both these problems.

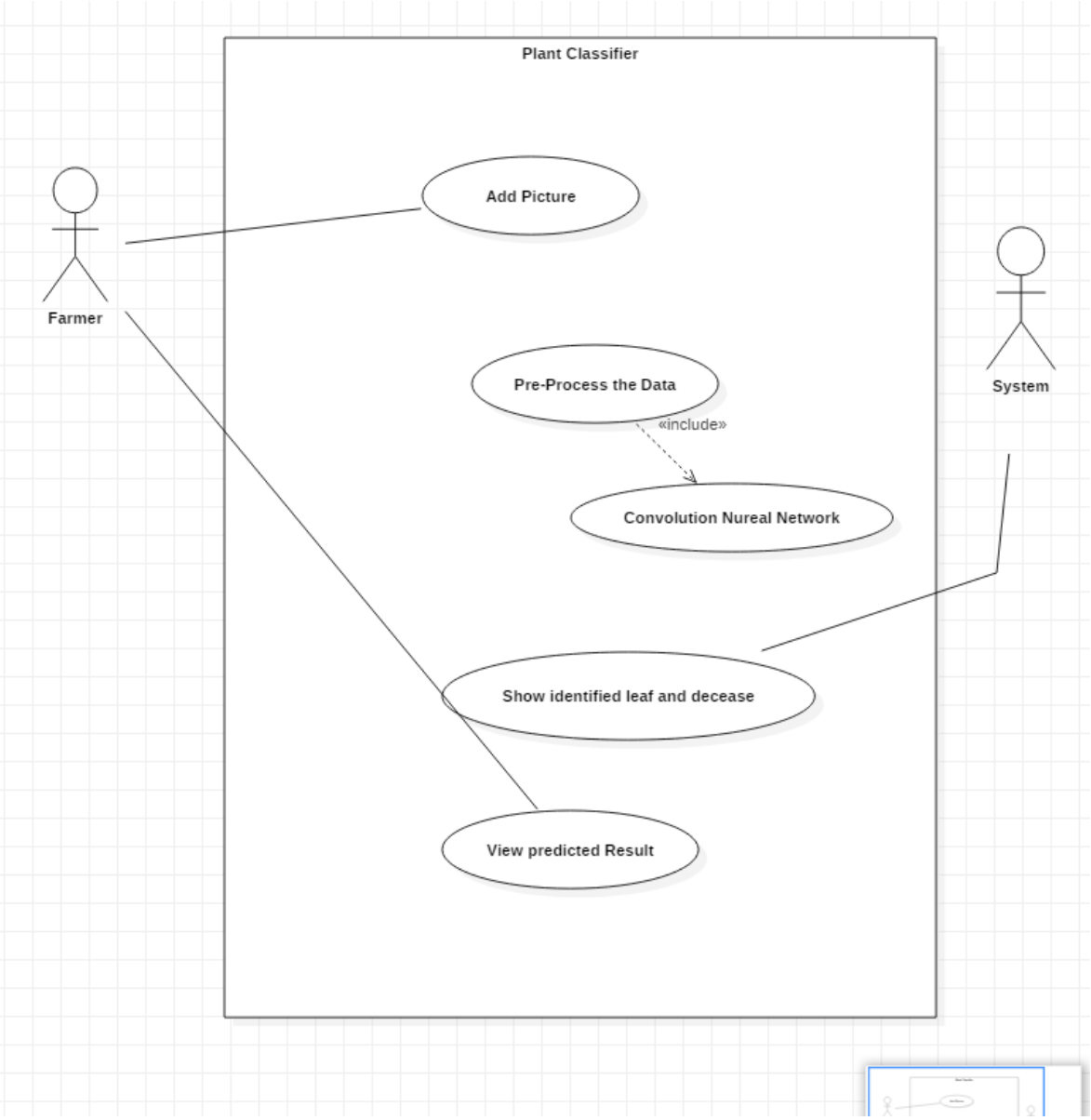
ResNet50 is a 50 layer Residual Network. There are other variants like ResNet101 and ResNet152 also.

## 3.4 Data Module

There are 34 data classes for train the model, and after training it will test the data for compare if it has trained correctly or not.

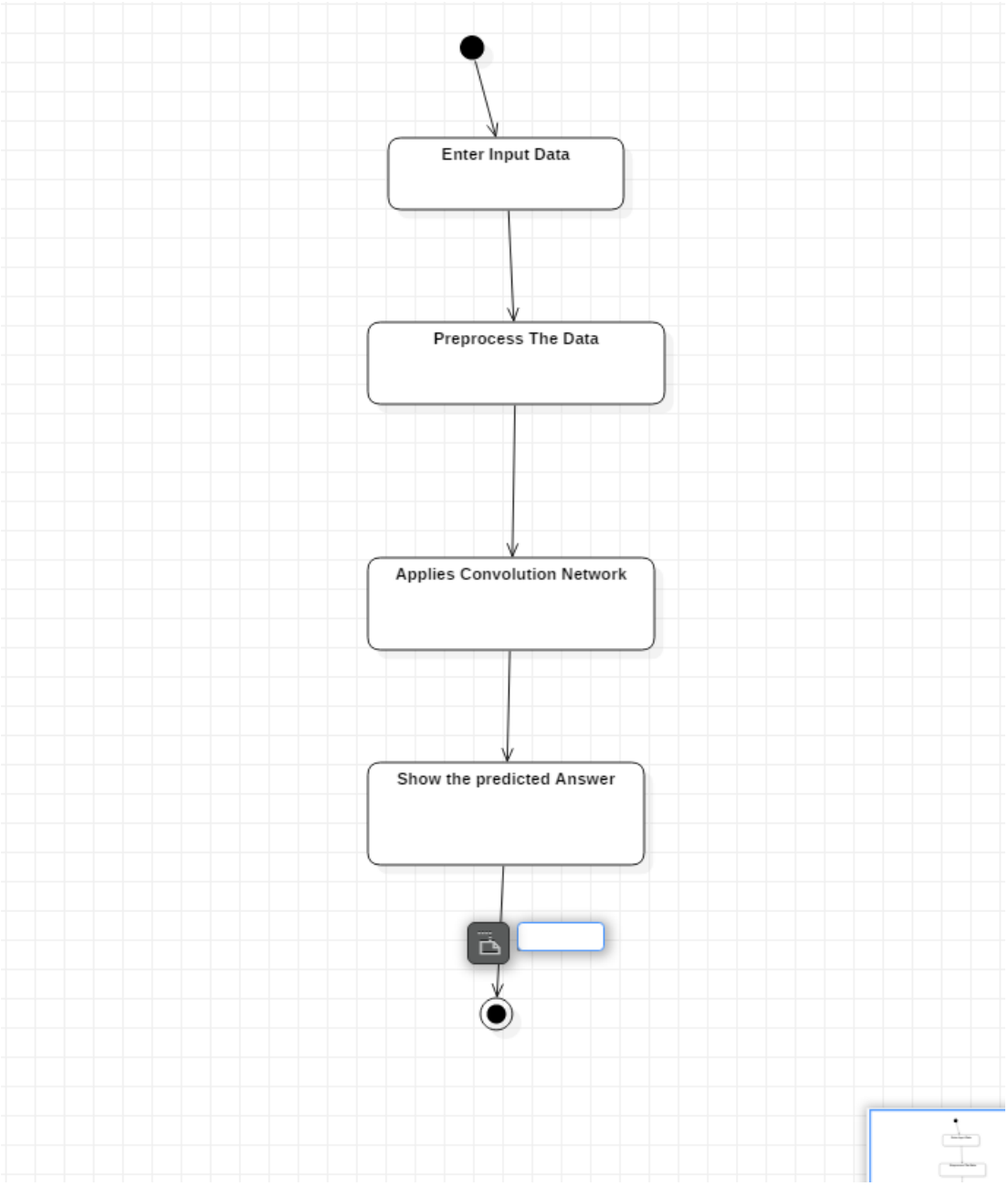
# CHAPTER 4 : UML DIAGRAMS

## 4.1 UseCsase Diagram




*UseCase Diagram*

## 4.2 Activity Diagram




*Activity Diagram*

## CHAPTER 5 : IMPLEMENTATION DETAILS

The model uses convolution neural network with transfer learning. Resnet50 model is used in the transfer learning module. The dataset is similar to imagenet dataset used by resnet50 for training so freezing first 80% layers and training is only done last 20% layers.

Resnet50 model is downloaded from keras without the top layer and the weights of imagenet dataset. After downloading according to the requirement last 3 layers that includes Global Averaging layer,Dense layer and an output layer specifying 34 classes are added to the model.

Previously trained model:

```
1
Found 35728 images belonging to 34 classes.
Found 8929 images belonging to 34 classes.
5
178  number of layers
C:\Users\Darshil Chikani\Anaconda3\lib\site-packages\tensorflow\python\keras\engine\training.py:1844: UserWarning: `Model.fit_g
enerator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/2
1116/1116 [==============================] - 7004s 6s/step - loss: 2.3595 - accuracy: 0.4203 - val_loss: 0.5644 - val_accuracy:
0.8741
Epoch 2/2
1116/1116 [==============================] - 6949s 6s/step - loss: 0.5210 - accuracy: 0.8851 - val_loss: 0.2587 - val_accuracy:
0.9311
Saved model to disk
```

## CHAPTER 6 : APPENDIX

### REFERENCES


1. http://cs231n.github.io/convolutional-networks/

2. https://machinelearningmastery.com/transfer-learning-for-deep-learning/

3. DeCAF reported similar findings in 2013. The framework in this paper (DeCAF) was a Python-based precursor to the C++ Caffe library.

4. CNN Features off-the-shelf: an Astounding Baseline for Recognition trains SVMs on features from ImageNet-pretrained ConvNet and reports several state of the art results.

5. https://www.quora.com/What-is-the-deep-neural-network-known-as-%E2%80%9CResNet-50%E2%80%9D

6. https://keras.io/getting-started/functional-api-guide/#first-example-a-densely-connected-network

7. https://medium.com/data-science-group-iitr/building-a-convolutional-neural-network-in-python-with-tensorflow-d251c3ca8117

8. https://www.tensorflow.org/tutorials/estimators/cnn

9. https://machinelearningmastery.com/save-load-keras-deep-learning-models/

10. https://arxiv.org/abs/1511.08060

# CHAPTER 7: TECHNOLOGY AND TOOLS

## 7.1 Environment

- Python
- Python libreries: keras , matplotlib , sklearn , scipy, etc.

## 7.2 IDE

- Jupyter Notebook

## 7.3 Technology

- Convolution Neural network
- Transfer learning