**Plant Disease Detection App for Farmers**

**Group 1**

**Frank 1930026136**

**Connor 1930026139**

**Jorian 1930026170**

A report submitted to

Dr. Canhao XU

For

Neural Networks and Deep Learning (1002) (Dr. Canhao XU) [Semester 1 of 2022-2023]

Content

[1. Introduction 2](#_Toc122471855)

[2. Dataset 3](#_Toc122471856)

[3. Data Prepoccessing 5](#_Toc122471857)

[4. Methodology 6](#_Toc122471858)

[4.1 LeNet 6](#_Toc122471859)

[4.2 AlexNet 7](#_Toc122471860)

[4.3 Inception 11](#_Toc122471861)

[4.4 ResNet 12](#_Toc122471862)

[4.5 Siamese 17](#_Toc122471863)

[5. Conclusion 20](#_Toc122471864)

[6. Deployment on Android 20](#_Toc122471865)

[7. Reference 23](#_Toc122471866)

# Introduction

Like people, plants can contract a wide range of diseases. We can create an app that enables farmers to recognize plant health conditions and types of diseases by their thanks to the deep learning models we have learned in this course.

We want to make an Android application, which is based on the premise of neural network to judge the virus type of plant leaves. In the selection of the neural network, we have five choices, and have carried out multiple EPOCH practices. Of course, the scale of the data set is also worthy of attention. Our dataset has more than 24,000 images of plant leaf diseases.

After our visit and observation on the UIC campus, we found fallen leaves with similar characteristics in the fallen leaves by the lake. Therefore, we are more convinced that the feasibility of this project is excellent. Of course, this needs to be judged by the camera of the Android app, so our project is probably to judge the type of deciduous virus with the neural network model, import real-time pictures from the mobile phone, and analyze the results



Figure 1 Early Blight

Figure 2 Bacterial Spot

Figure 3 Late Blight

# Dataset

After our efforts to find and try to match our pursuit of this project, we finally found a data set named “PlantVillage” on the Kaggle website:

[*https://www.kaggle.com/datasets/emmarex/plantdisease*](https://www.kaggle.com/datasets/emmarex/plantdisease)

The content of the dataset is of diseased plant leaf images and corresponding labels. This data set has a total of more than 24,000 photos of plant leaf viruses, including 3 types of plants, tomato, potato, pepper, and a total of 15 categories of different diseased plants. The number of datasets is considerable.

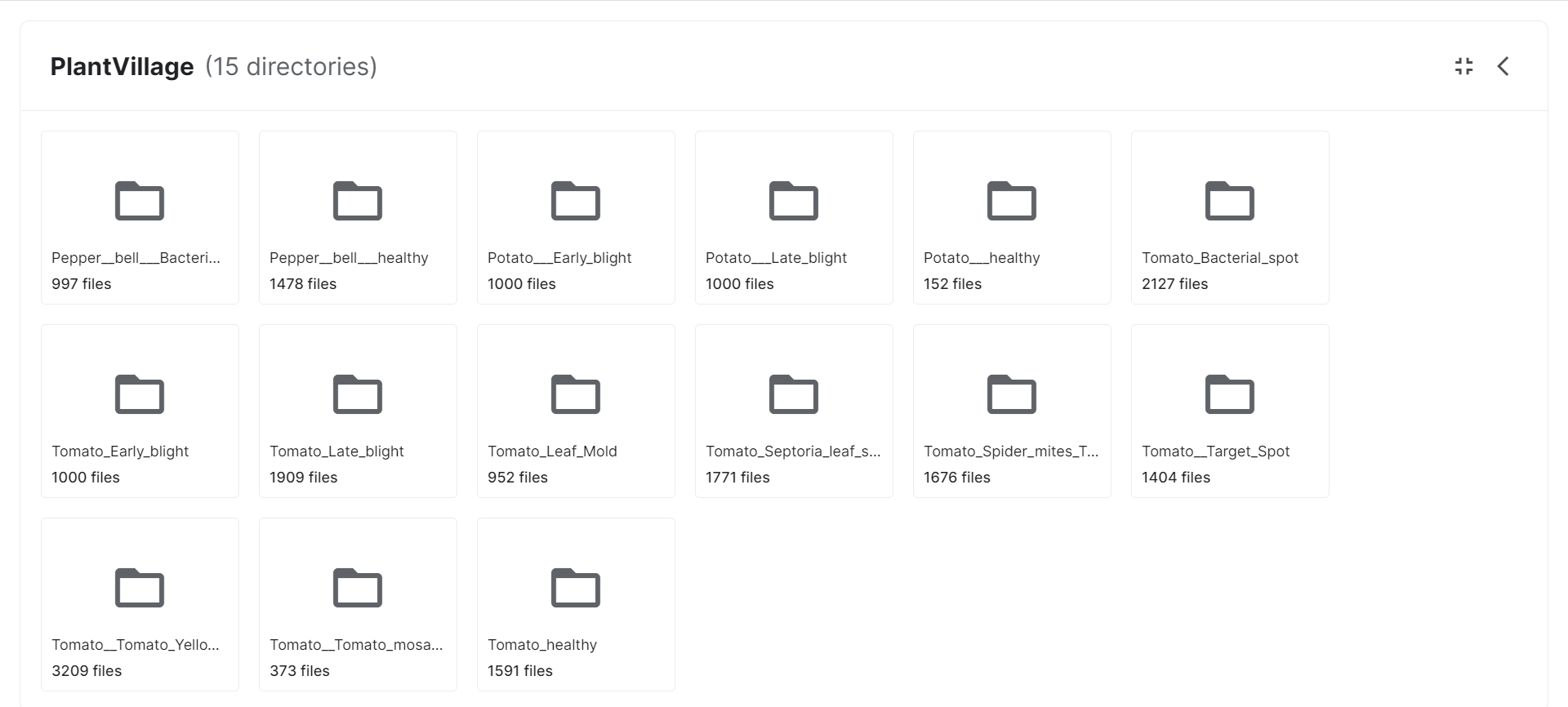
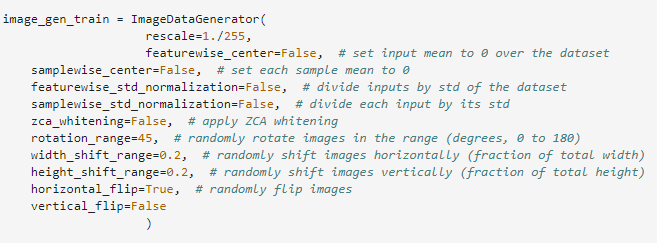


Figure 4 Dataset information

Of course, we must also emphasize the accuracy of this data set itself. After all, we are not students of biological sciences, and it is difficult to judge plant leaves directly from the eyes, but I am sure from the feedback of other users of this data set. The performance of this dataset is excellent and can be used completely.

# Data Prepoccessing

We set our ImageDataGenerator input mean and sample mean to 0, divide inputs by std of the dataset and also divide themselves by their own std, apply ZCA whitening, randomly rotate images in the range, shift them horizontally and vertically, and also flip them, as in the picture below:



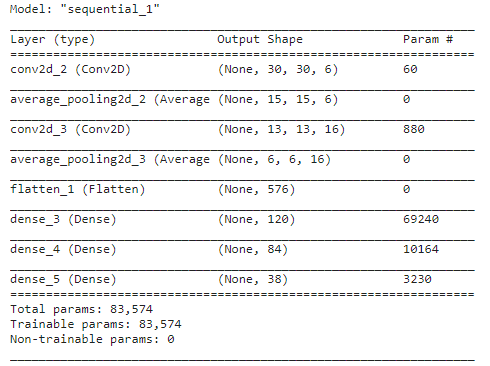
# Methodology

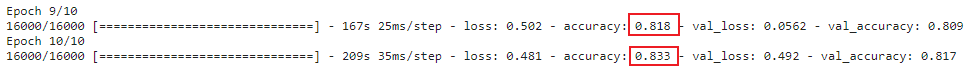
## 4.1 LeNet

LeNet is a classical convolutional neural network and one of the origins of modern convolutional neural networks. Yann used the network for the postal code recognition of the post office, and has good learning and recognition ability. LeNet, also known as LeNet-5, has one input layer, two convolution layers, two pooling layers, and three full connection layers (the last full connection layer is the output layer).

LeNet5 consists of seven layers, namely C1, C3, C5 convolution layer, S2, S4 downsampling layer (downsampling layer is also called pooling layer). F6 is a full connection layer, and the output is a Gaussian connection layer. This layer uses the softmax function to classify the output images. In order to correspond to the model input structure, the 28 \* 28 image in MNIST is expanded to 32 \* 32 pixels. Each layer is described in detail below. The C1 convolution layer is composed of 6 convolution kernels of different types with a size of 5 \* 5. The step size of the convolution kernel is 1, and there is no zero filling. After convolution, six 28 \* 28 pixel feature maps are obtained; S2 is the maximum pooling layer. The size of pooling area is 2 \* 2, and the step size is 2. After S2 pooling, six 14 \* 14 pixel feature maps are obtained; C3 convolution layer is composed of 16 different convolution kernels with a size of 5 \* 5. The step size of the convolution kernel is 1, and there is no zero filling. After convolution, 16 characteristic images with a size of 10 \* 10 pixels are obtained; S4 is the largest pooling layer. The size of pooling area is 2 \* 2, and the step size is 2. After S2 pooling, 16 feature maps with the size of 5 \* 5 pixels are obtained; The C5 convolution layer consists of 120 different convolution kernels with a size of 5 \* 5. The step size of the convolution kernel is 1, and there is no zero filling. After convolution, 120 characteristic images with a size of 1 \* 1 pixel are obtained; 120 characteristic maps with the size of 1 \* 1 pixel are spliced together as the input of F6, which is a fully connected hidden layer composed of 84 neurons. The activation function uses sigmoid function; The final output layer is a softmax Gaussian connection layer composed of 10 neurons, which can be used for classification tasks.

And this is our process:





## 4.2 AlexNet

AlexNet carries forward LeNet's ideas and applies the basic principles of CNN to a very deep and wide network. AlexNet mainly uses the following new technologies:

ReLU was successfully used as the activation function of CNN, and its effect was verified to exceed that of sigmoid in deeper networks, which successfully solved the gradient dispersion problem of sigmoid in deeper networks. Although ReLU activation function was proposed a long time ago, it was not developed until the emergence of AlexNet.

Dropout is used to randomly ignore some neurons during training to avoid over fitting the model. Although Dropout has been discussed in a separate paper, AlexNet has put it into practice and proved its effectiveness through practice. In AlexNet, Dropout is mainly used in the last full connection layers.

Use overlapping maximum pooling in CNN. Before that, CNN generally used average pooling, and AlexNet all used maximum pooling to avoid the fuzziness effect of average pooling. In addition, AlexNet proposes to make the step size smaller than the size of pooled core, so that there will be overlap and coverage between the outputs of pooled layers, which improves the richness of features.

The LRN layer is proposed to create a competition mechanism for the activities of local neurons, which makes the values with larger responses become larger, and inhibits other neurons with smaller feedback, thus enhancing the generalization ability of the model.

CUDA is used to accelerate the training of deep convolution network, and GPU's powerful parallel computing ability is used to handle a large number of matrix operations during neural network training. AlexNet uses two GTX 580 GPUs for training, and a single GTX 580 has only 3GB video memory, which limits the maximum size of the trainable network. Therefore, the author distributes AlexNet on two GPUs, and stores half of the neurons' parameters in the video memory of each GPU. Because the communication between GPUs is convenient and they can access the video memory without using the host memory, it is also very efficient to use multiple GPUs at the same time. At the same time, AlexNet's design allows the communication between GPUs to be carried out only at certain layers of the network, controlling the performance loss of communication.

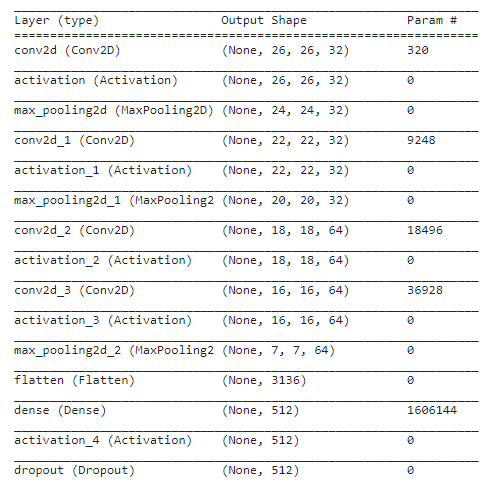
Data enhancement: randomly intercept 224 \* 224 areas (and horizontally flipped images) from 256 \* 256 original images, which is equivalent to 2 \* (256-224) ^ 2=2048 times more data. If there is no data enhancement, CNN with many parameters will fall into over fitting only depending on the original data volume. The use of data enhancement can greatly reduce the over fitting and improve the generalization ability. When making predictions, take four corners of the picture plus five positions in the middle, and flip left and right to get a total of 10 pictures, predict them and calculate the average of the 10 results.

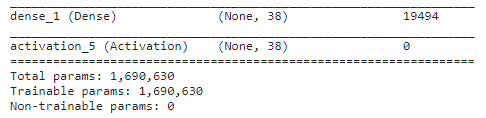
Alexnet also has its special characteristics. ReLU based deep convolution network is several times faster than tanh and sigmoid based network training.

After ReLU is used, it will be found that the values after the activation function do not have a range like the tanh and sigmoid functions. Therefore, a normalization is generally performed after ReLU. LRU is a steady way to propose a method. In neuroscience, there is a concept called "Final intrusion", which refers to the influence of active neurons on its peripheral neurons.

Dropout is also a frequently mentioned concept, which can effectively prevent the over fitting of neural networks. Compared with the general linear model, the regular method is used to prevent the model from over fitting. In the neural network, Dropout is implemented by modifying the structure of the neural network itself. For neurons at a certain layer, delete some neurons randomly through the defined probability, while keeping the number of neurons at the input layer and output layer unchanged, and then update the parameters according to the learning method of the neural network. In the next iteration, delete some neurons randomly again until the end of the training.

And this is our process:



## 4.3 Inception

Inception network is an important milestone in the development history of CNN classifier. Before Inception, most popular CNN just stacked more and more convolutional layers to make the network deeper and deeper, hoping to get better performance.

The biggest feature of GoogLeNet is the use of the Inception module. Its purpose is to design a network with good local topology, that is, to perform multiple convolution operations or pooling operations on the input images in parallel, and to splice all the output results into a very deep feature map. Because different convolution operations and pooling operations, such as 1 \* 1, 3 \* 3 or 5 \* 5, can obtain different information of the input image, parallel processing of these operations and combining all results will obtain better image representation.

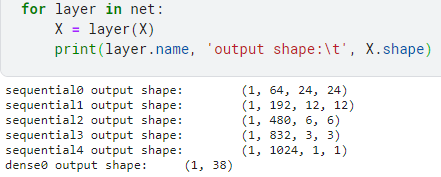
Compared with the sequential connection structure (such as VGG network) of the convolution layer and pooling layer, the inception structure has the following improvements:

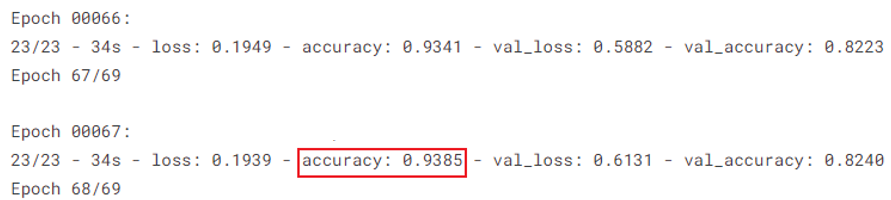
One layer of blocks includes 1x1 convolution, 3x3 convolution, 5x5 convolution, and 3x3 pooling (this size is not necessary, and can be adjusted as needed). In this way, each layer of the network can learn the characteristics of "sparse" (3x3, 5x5) or "non sparse" (1x1), which not only increases the width of the network, but also increases the adaptability of the network to the scale;

Use deep concat to synthesize features after each block to obtain nonlinear attributes.

Increasing the depth of the network according to this structure can improve the performance, but it also faces the problem of large computation (many parameters). In order to improve this phenomenon, Inception uses the idea of Network in Network for reference, and uses 1x1 convolution kernel to realize dimension reduction (which also indirectly increases the depth of the network), so as to reduce the network parameters:

And this is our process:





## 4.4 ResNet

ResNet aims to solve the "degradation" problem of deep neural networks.

Gradually overlay layers on the shallow network, and the performance of the model on the training set and test set will become better, because the model has higher complexity, stronger expression ability, and can better fit the potential mapping relationship. "Degeneration" refers to the situation where the network performance declines rapidly after adding more layers.

It is reasonable to stack more layers on the network. The solution space of the shallow network is included in the solution space of the deep network. The solution space of the deep network is at least as good as the solution of the shallow network, because only the added layers need to be changed into an identity map, and the weights of other layers can be copied intact from the shallow network, and the same performance can be obtained as the shallow network.

Obviously, this is an optimization problem, reflecting that the optimization difficulty of models with similar structures is different, and the difficulty growth is not linear. The deeper the model, the more difficult it is to optimize.

There are two solutions. One is to adjust the solution method, such as better initialization, better gradient descent algorithm, etc; The other is to adjust the model structure to make the model easier to optimize - changing the model structure actually changes the form of the error surface.

The author of ResNet starts with the latter to explore a better model structure. The stacked layers are called a block. For a block, the function that can be fitted is F (x). If the expected potential mapping is H (x), instead of letting F (x) directly learn the potential mapping, it is better to learn the residual H (x) − x, that is, F (x):=H (x) − x, so that the original forward path becomes F (x)+x, and F (x)+x is used to fit H (x). The author thinks that this may be easier to optimize, because it is easier to learn F (x) to be 0 than to learn F (x) to be an identity map - the latter can be easily realized through L2 regularization. In this way, for redundant blocks, identity mapping can be obtained by F (x) → 0, with no performance degradation.

The block formed by F (x)+x is called a ResidualBlock, that is, a residual block. As shown in the figure below, several similar ResidualBlocks are connected in series to form ResNet.

residual block has two paths F (x) and x, and the F (x) path fits the residual, which can be called the residual path. The x path is identity mapping identity mapping, which is called "shortcut". The ⊕ in the figure is an element wise addition, which requires that the dimensions of F (x) and x involved in the operation should be the same.

In the original paper, residual paths can be roughly divided into two types, one with bottomleneck structure, namely 1\*1 accumulative layer in the right of the figure below, which is used to reduce dimensions first and then increase dimensions, is called "bottomleneck block" mainly for the practical consideration of reducing computational complexity, and the other is called "basic block" without bottomleneck structure, as shown on the left of the following figure. The basic block consists of two 3 × 3 volumes of accumulative layers, bottomleneck block is composed of 1 × 1.

ResNet is the concatenation of multiple Residual Blocks. The design of ResNet has the following characteristics:

Compared with plain net, ResNet has many more "bypasses", that is, shortcut paths. The layers circled at the beginning and end of ResNet form a ResidualBlock;

In ResNet, all Residential Blocks have no pooling layer, and downsampling is implemented through the stream of conv;

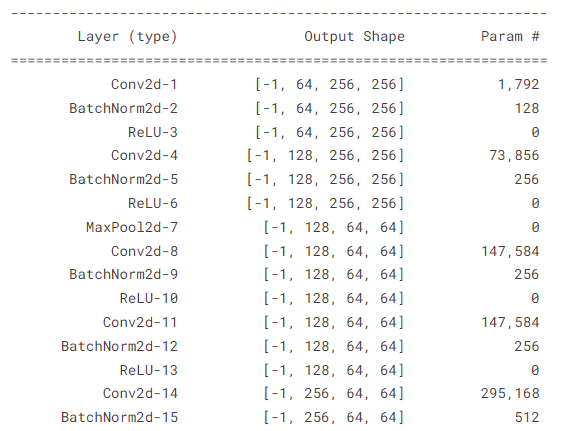
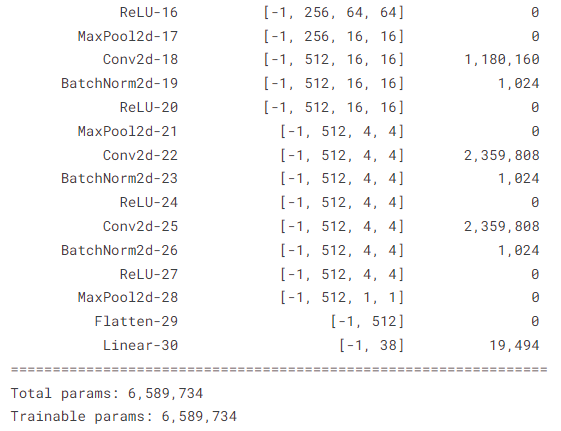
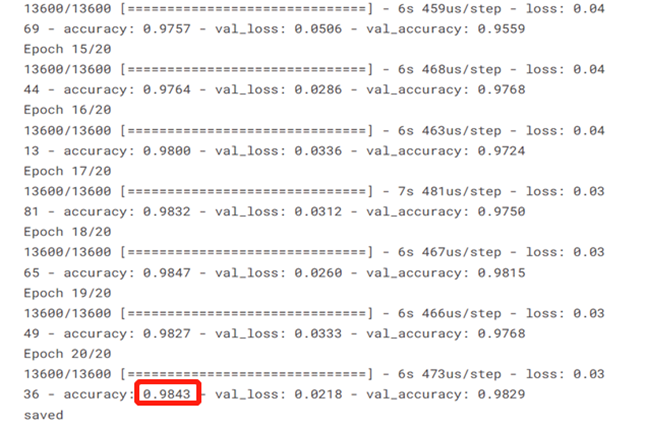
Respectively in conv3\_ 1、conv4\_ 1 and conv5\_ 1 ResidualBlock: reduce the sampling by one time, and increase the number of feature maps by one time, as shown in the block delineated by the dotted line in the figure;

Get the final features through Average Pooling, rather than through the full connection layer;

Each convolution layer is followed by the BatchNorm layer, which is not marked in the figure for simplification;

The ResNet structure is very easy to modify and expand. By adjusting the number of channels in the block and the number of stacked blocks, you can easily adjust the width and depth of the network to obtain networks with different expression capabilities, without worrying too much about the "degradation" of the network. As long as the training data is sufficient and the network is gradually deepened, you can obtain better performance.

And this is our process:

## 4.5 Siamese

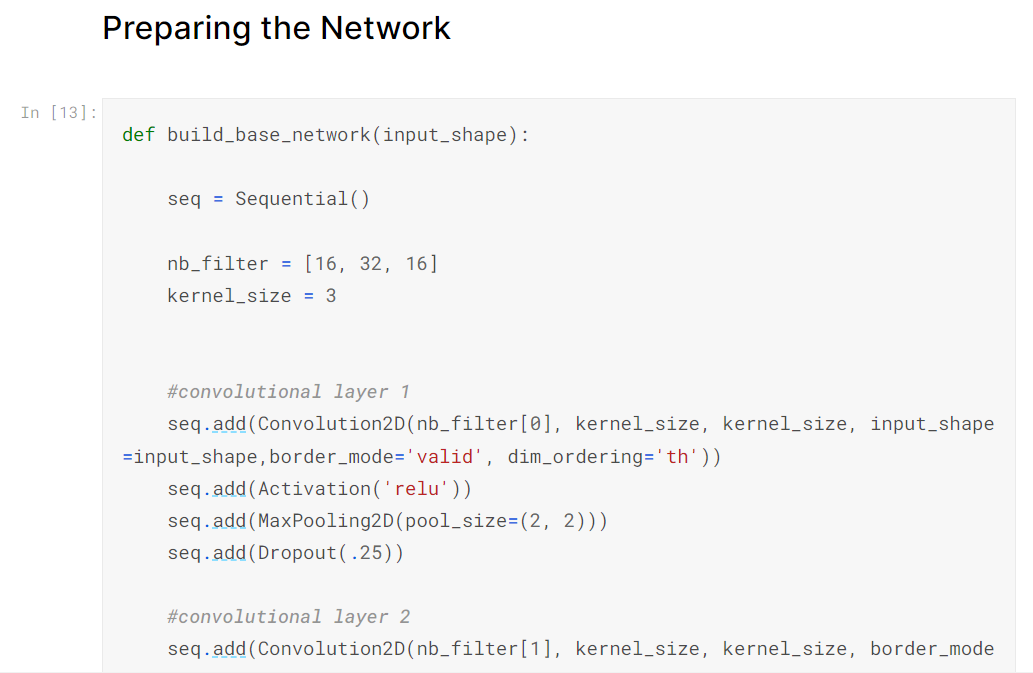
Siamese uses the same convolutional neural network to extract features from two input images, and then judges whether they belong to the same category by comparing the distance between their feature vectors.

Prepare two datasets. The positive sample is used to tell the neural network what the same category is. The positive sample sampling is to randomly select two picture combinations from the same category, and then label 1 to indicate a positive sample. A negative sample represents the difference between two images. One image is extracted from the data set each time, and then one image is randomly selected from different categories and combined with it to form a negative sample. Then the label is 0, and 0 means the similarity is 0.

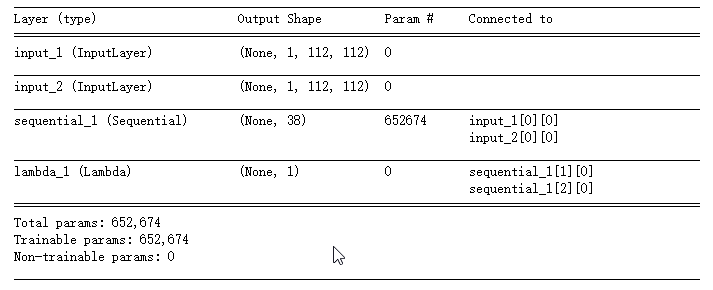
The Siamese model is mainly composed of two parts. The first part is the full connection layer and a full connection layer for feature extraction. The second full connection network is used to compare the distance between the extracted vectors of the convolutional neural network.

First, a convolution neural network is designed to extract image features. Input two pictures into the neural network each time. Note that this is the same neural network. Then each image passes through this neural network to get a feature vector. Then subtract the two eigenvectors to get a new vector. Input the new vector obtained by subtraction to a full join layer to obtain a scalar. Finally, use the sigmoid function. If two pictures are of the same category, the output should be close to 1. On the contrary, the output should be close to 0.

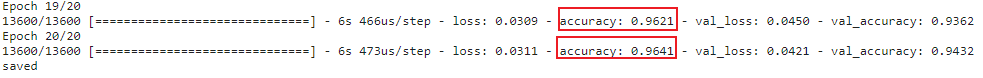
For the category of an image to be predicted, the same number of images can be extracted from different category weeks, and then the image to be predicted can be input into the twin neural network for prediction. The prediction is a number between 0 and 1. The image to be predicted will belong to the category by calculating which of the images of different categories it is similar to.



The summary of the model:



Training process and accuracy:



# Conclusion

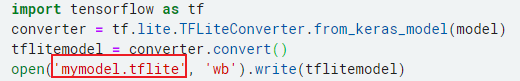
After training the six models, we compared the acuracy of them:

The top 1 is ResNet, which is 98.43%. the Siamese Network and Inception also work well, which are above 90%. However, the others are not satisfatory.

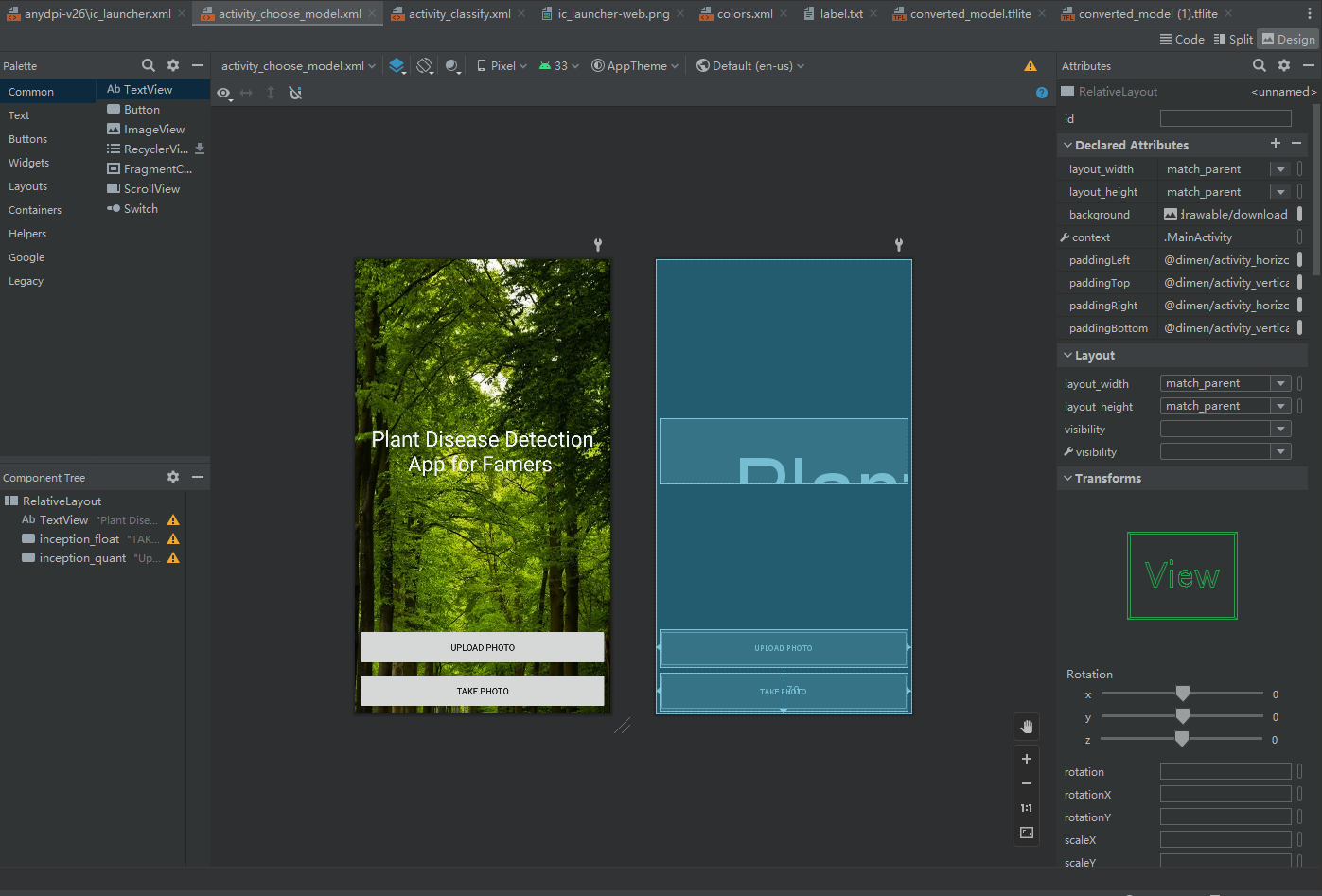
|  |  |
| --- | --- |
|  | **Accuracy** |
| **ResNet** | **98.43%** |
| **Siamese Network** | **96.41%** |
| **Inception** | **93.85%** |
| **AlexNet** | **88.37%** |
| **CNN** | **86.33%** |
| **LeNet** | **83.30%** |

# Deployment on Android

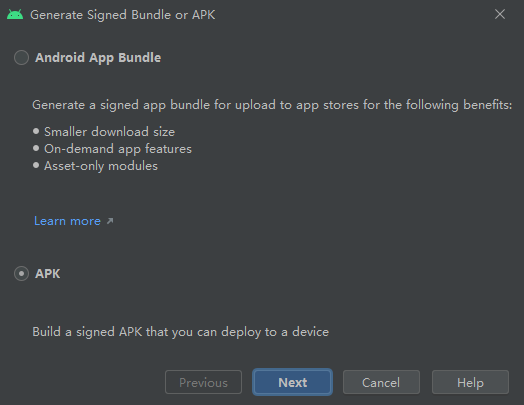
Because ResNet works the best, we converted it into tflite model file:



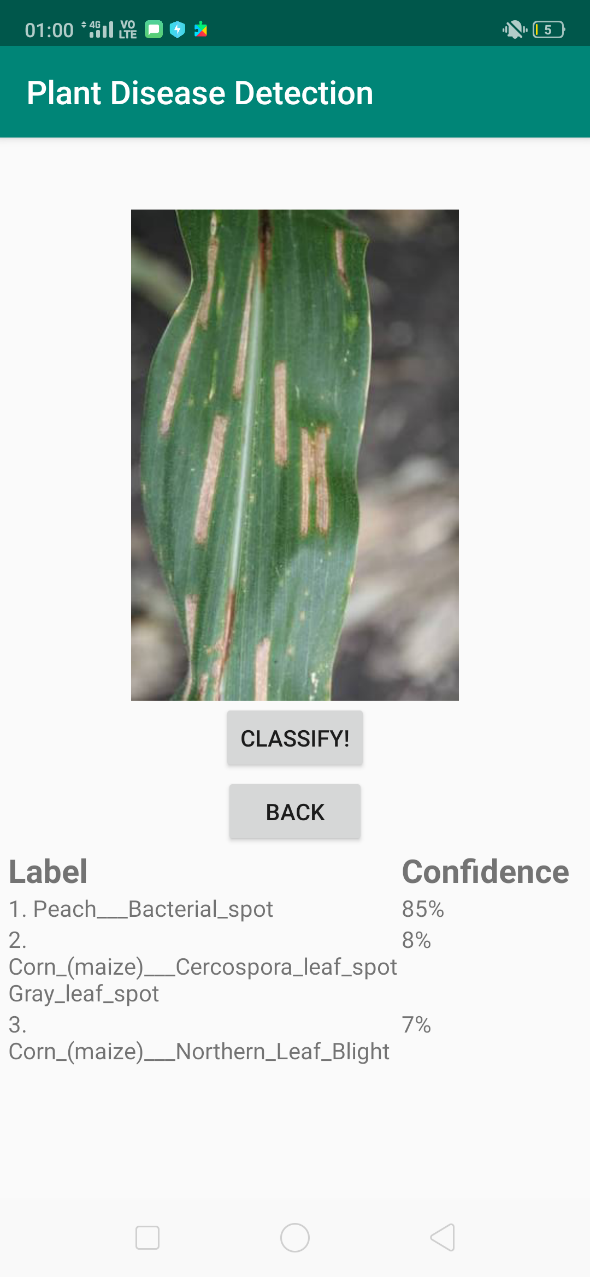
Then we developed an Android app on Android Studio:



After that, we generated the APK and installed it on our Android phone:



The user interface is like the following:

# Reference

Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). Using deep learning for image-based plant disease detection. Frontiers in plant science, 7, 1419.

Mohameth, F., Bingcai, C., & Sada, K. A. (2020). Plant disease detection with deep learning and feature extraction using plant village. Journal of Computer and Communications, 8(6), 10-22.

Wspanialy, P., Brooks, J., & Moussa, M. (2020). An image labeling tool and agricultural dataset for deep learning. arXiv preprint arXiv:2004.03351.

Syamsuri, B., & Kusuma, G. P. (2019). Plant disease classification using lite pretrained deep convolutional neural network on android mobile device. 2019 International Journal of Innovative Technology and Exploring Engineering (IJITEE).

Shoaib, M., Ahmad, F., & Rehman, M. A. (2022). Deep Learning-Based Plant Disease Detection Using Android Apps. In Artificial Intelligence Applications in Agriculture and Food Quality Improvement (pp. 148-168). IGI Global.