Research on Disease Detection Systems in Tomato Plants using Artificial Intelligence

Nguyen Le Dung, PhD
Department of Automation and Control,
Faculty of Electrical and Electronics Engineering
Ho Chi Minh City University of Technology,
VNU-HCM, Viet Nam

Tran Hau Van Toan
Department of Automation and Control
Faculty of Electrical and Electronics Engineering
Ho Chi Minh City University of Technology
VNU-HCM, Viet Nam

ABSTRACT
This working personally discusses how to approach the Raspberry Pi 3B + Board in combination with the 8MP Raspberry Pi V2 camera, and at the same time about algorithm knowledge with the Raspberry Pi 3B + board, and also be able to communicate with reduction motors. V1 speed, RC servo motor, H bridge circuit.

During the implementation of the individual use object detection method to perform identification and detection of tomato disease. Combine with a self-driving car with a camera to make it possible to sample tomato leaf disease in the most realistic way.

General Terms
Raspberry Pi 3B +, Object Detection

Keywords
Raspberry Pi 3, Yolo

1. INTRODUCTION
At present, industrialization and modernization are developing day by day. In which, automation is one of the indispensable fields in the industry and agriculture. It contributes to increase production productivity, and save labor costs. Today artificial intelligence is increasingly being applied to the modernization industry, applying artificial intelligence to perform complex tasks more efficiently. From those practices, the article identifies and detects tomato diseases by artificial intelligence methods to help farmers easily recognize the condition as well as the disease of the plant, to provide timely solutions for farmers to treat diseases for plants to contribute to increasing yield and quality of crops.

Currently, agriculture is one of the important areas in the economic development of the country. In response to maximizing the benefits from agriculture, domestic and foreign researchers need to find solutions to improve productivity. One of the factors affecting the yield of crops is pests and diseases. In the face of the serious impact of pests and diseases on the quality and productivity of crops, scientists have had practical and effective solutions, among which are applications using artificial intelligence.

Artificial intelligence is no longer an unfamiliar field in our life today, it is widely applied in fields such as transportation, manufacturing, healthcare, education, communication. Especially, it is applied in agriculture such as using artificial intelligence for harvesting robots, crop monitoring robots, ... It helps people save time, effort, improve quality. amount. Using artificial intelligence in agriculture makes it possible for people to promptly detect the condition of plants.

Fig. 1-1: The robot detects the condition of the crop

Although there are many differences in the structure of robots, current research focuses on service applications and operation of robots in natural environments. With the development of society and modernization in countries around the world, many new services are formed that change the perspective of robots, from robots for industries to robots for agriculture. . On the other hand, humans also apply robots in combination with artificial intelligence to apply to agriculture, in order to improve productivity efficiency, improve the quality of crops as well as products. In this research paper, we present a simple robotic system, or we could call this simple system a self-propelled vehicle that detects and identifies a tomato plant disease. The system uses artificial intelligence to detect pests and diseases in tomato plants.

1.1 Research purposes
Learn how and classify a specific subject (tomato plant leaf disease). Choosing the appropriate method for the requirements. In order to apply these methods, it is necessary to design and build hardware systems in combination with research and development algorithms, thereby assessing the operability and accuracy of the model in use, helping the system accurately detects pests and diseases.

1.2 Target
The most important goal of this paper is to detect and identify tomato plant diseases through the leaves of the plant and deploy on self-propelled vehicle for experiment, which includes the following main goals:

- Understand the problem of object detection and recognition of deep learning.
- Selecting models and algorithms suitable for the research topic.
- Deploying models on hardware for real experimentation.

1.3 Methods of implementation

To build a system of self-propelled vehicles to detect and identify diseases in tomato plants, we need to perform the following steps:

The first is to learn about methods for accuracy when performing object detection and recognition in machine learning, so that we can choose methods that suit the identification and disease detection requirements of tomato plants. leaf.

Second, we need to understand the method of object detection and recognition, also known as full object detection, using deep learning.

Thirdly, to select a model for disease detection and identification of tomato plants with a small size, real-time speed and moderate accuracy to be able to perform in detection and identification models disease on tomato plants.

Fourth, learn about hardware components suitable for the model such as sensors, DC geared motors, RC servos, control circuits ...

Fifthly, implementing a combination of hardware and software together to test the model in a real environment.

Sixthly, to improve the accuracy of the model.

1.4 Selection of tomato plant diseases to detect

The red ripe tomatoes are an indispensable daily food in any family, tomatoes not only have the effect of preventing and treating a number of diseases such as anemia, good cardiovascular health, digestive system stability, but also Used a lot in beauty because it is good for the skin, for the hair. It has the ability to make your whole body good and beautiful.

Due to such a large demand for tomatoes, but the supply of tomatoes is still lacking, especially in the off season, tomatoes that are not in the season are difficult to take care of, especially many diseases that make tomatoes worse and fruit quality is not. as expected, the fruit is small, not smooth. The most important step in taking care of tomato plants to have good productivity and quality is to prevent diseases for plants. Disease in tomato plants is one of the main reasons affecting the development of fruit yield and quality of tomatoes when harvested, tomato plants suffer from many different diseases.

Common diseases that affect the yield of tomato plants include early chlorosis (ring spot), leaf spot disease, leaf yellow spot fungus, bacterial foliar spot disease, helix disease, and mildew, yellow wilt disease, anthracnose…. These are typical diseases that affect the quality and yield of tomatoes.

2. TRAINING IN DISEASE DETECTION AND IDENTIFICATION MODELS

2.1 Use deep learning for research

Plant diseases are one of the most serious and lasting problems in the development of agriculture worldwide. Early detection and proper treatment are important steps in efforts to increase crop yield and yield. There are several ways to analyze plant diseases including visual examination by specialists, biological examinations or automated computer-based diagnostic systems. The problem with specialist visual inspection and biological examinations is that such analyzes are often time consuming, costly, and do not identify diseases in time. In this context, many methods of diagnosis based on artificial intelligence capable of quickly and reliably identifying diseases have been proposed.

Currently, researchers on plant diseases have published out to the community a data set of diseased leaves including many different looking plants such as apples, strawberries, cherries, oranges, potatoes, tomatoes, ... Data set with thousands of diseased and healthy leaf samples from a variety of plants. This data set, called Plantvillage, has been published and is intended for AI researchers to participate in a disease classification contest to see who produces the best results.

Reference from the documents of researchers about artificial intelligence [1]. We see that they use CNN network to apply the problem of disease classification. The accuracy of using CNN networks gives positive results that can be applied to my topic. However the data in [1] used is the PlantVillage dataset, the disadvantage of this dataset is that most of the images were taken in the laboratory (ie each leaf was taken on a uniform background), no must be in real terms. In order to promote the strength of CNN in the personal classification problem, they have actively collected data from many sources on the internet, the number of which are photos taken from many different places (collected directly to the garden but only leaves healthy).

CNN network is also considered as deep learning as a high-level method of machine learning developed and researched by scientists. Currently, deep learning is one of the methods that are widely used in practice such as self-driving cars, healthcare industry, text recognition, and especially in agriculture, it is used for the spread of diseases of plants, planting, harvesting,….

CNN models of deep learning were created by researchers and created such as Alexnet, Inception, VGG, Resnet. It goes far beyond the accuracy of the Support Vector machine (SVM) methods.

CNN carriers such as Alexnet, Inception, VGG, Resnet, Mobilenet are mainly used to classify objects. Because the topic was the detection and identification of tomato plant disease, we learned about other CNN networks that mistakenly extracted the location of the object to be classified. CNN networks extract object locations such as R-CNN, SSD, Faster R-CNN.

We see Faster R-CNN as a powerful model of R-CNN. The SSD model is designed for object detection in real time. Faster R-CNN uses the site suggestion network to create boundary boxes and uses those boxes to classify objects. Although considered the correct start, the entire process ran at 7 frames per second. Much lower than what is needed for a real-time processing need. SSD speeds up the process by eliminating the need for area proposal networks. To deal with the reduced accuracy problem, SSD applies several enhancements including multi-size feature maps and use of default boxes. These enhancements allow the SSD to come close to the accuracy of Faster R-CNN but be able to use lower resolution images, resulting in higher speeds. Therefore, the SSD is chosen by us to apply in our research articles.

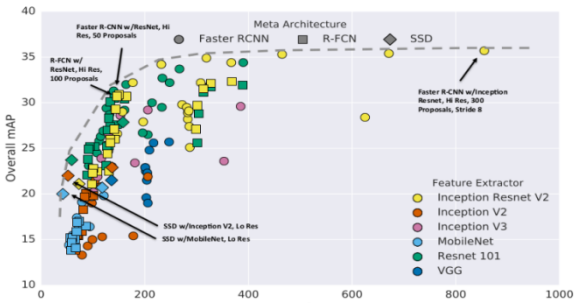

Fig. 2-1: Model accuracy over time

Using the above image from the research paper [6], we selected the SSD model combined with Mobilenet to perform the problem of tomato plant disease detection and classification, based on the data of the graph, We can see that the SSD-mobilenet model has real-time fast speed, relatively stable accuracy, suitable for the research of the nuclei.

Below is the CNN network model of SSD-Mobilenet.

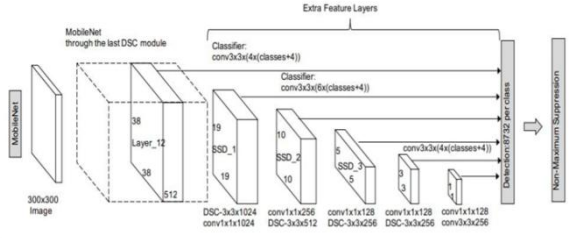

Fig. 2-2: SSD-mobilenet network architecture diagram

The SSD relies on a standard architecture (Mobilenet) propagation process to generate an early 3 dimensional feature map output block. We call this network architecture the base network (from input Image to SSD_1). We will then add the structures behind the base network to perform object recognition as part of the Extra Feature Layers in the diagram. These layers are simply explained as follows:

The layers of the SSD model:
Input Layer: Receive as input of images with size (width x height x channels) = 300 x 300 x 3 for SSD300 architecture or 500 x 500 x 3 for SSD500 architecture.

Mobilenet: It's a base network that uses Mobilenet's architecture but removes some layers fully connected at the end. The output of this layer is Layer_12 and is a feature map of size 38 x 38 x 512.

Layer_12: We can consider Conv4_3 as a feature map with the dimensions 38 x 38 x 512. On this feature map we will apply 2 main transformations: Apply a convolutional layer as a normal CNN network to obtain output next layer. Specifically, the convolutional layer has a 3 x 3 x 1024 convolutional kernel, the output obtained SSD_1 has a size of 19 x 19 x 1024. At the same time, we also apply a classifier (as shown in the diagram). on the 3 x 3 convolutional filter to identify the object on the feature map. This is a rather complicated process as it has to ensure object detection (through bounding box detection) and object classification. First we will divide the feature map of size 38 x 38 x 512 into a grid cell of size 38 x 38 (ignore the depth because we will do convolution over all depth). Then each cell on the grid cell will create four default bounding boxes with different aspect ratios, for each default bounding box we need to find the following parameters: the distribution of the label's probability to be a vector the string has n_classes + 1 dimension (Note the number of classes always plus 1 for the background). At the same time we need to add 4 parameters offsets to define the bounding box of the object in the frame. So on a default bounding box there will be n_classes + 4 parameters and on a cell there will be 4 * (n_classes + 4) outputs to be forecasted. Multiply by the number of cells of Conv4_3 to get the output of a tensor of size 38 x 38 x 4 x (n_classes + 5), in case of considering background as 1 label, the tensor is 38 x 38 x 4 x (n_classes + 4). And the number of bounding boxes produced is 38 x 38 x 4.

The process of applying classifier to feature map is similar with layers SSD_1, SSD_2, SSD_3, SSD_4, SSD_5. The shape of the following layers will depend on how the convolutional process is applied to the previous layer, the size of the kernel filter (as shown in the diagram above, kernel_size is always 3 x 3) and the stride (size of the jump) of convolution. On each cell of the feature map we define a set of 4 or 6 default bounding boxes. Therefore, the number of default boxes spawned in the next layers is as follows:

SSD_1: 19 × 19 × 6 = 2166 boxes (6 boxes / cell)

SSD_2: 10 × 10 × 6 = 600 boxes (6 boxes / cell)

SSD_3: 5 × 5 × 6 = 150 boxes (6 boxes / cell)

SSD_4: 3 × 3 × 4 = 36 boxes (4 boxes / cell)

SSD_5: 1 × 1 × 4 = 4 boxes (4 boxes / cell) Total number of boxes at output will be: 5776 + 2166 + 600 + 150 + 36 + 4 = 87325776 + 2166 + 600 + 150 + 36 + 4 = 8732. That is, we need to predict the class for about 8732 frames on the output. The data on the layers above are referenced and cited from the document [10].

2.2 Tensorflow Libraries and Google Colab

2.2.1. Tensorflow Library

TensorFlow is an open source library developed by Google to support the research and development of AI (Machine learning / Deep learning) applications. The first version of Tensor Flow was announced in November 2015 and Google recently announced version TensorFlow 2.0 Alpha on March 4, 2019. The Core part of TensorFlow is written in C ++ but users can use either C ++ or Python to develop their application (python is still the most used language for writing TensorFlow).

The name TensorFlow is derived by combining Tensor + Flow (flow). Here, Tensor is a multidimensional array data type. Therefore we can understand simply, TensorFlow is the stream or data stream of Tensor. Built in many machine learning libraries.

Good compatibility and expansion. Developed by Google for machine learning for both research and construction of real-world applications. Popularity and much support for users.

From the above objective factors, we can access deep learning through a published CNN network, so we chose tensorflow to perform the training of object detection and classification model. use SSD model mobilenet.

2.2.2. Google Colab

To write programs using TensorFlow, we use Python IDE, for

example PyCharm (jointly developed by JetBrains, so has the same interface as IntelliJ), Jupyter Notebook or Atom, ... , in this series of tutorials, we will be using Google Colab since adopting Machine Learning / Deep Learning often requires the system to have high speed and processing power (usually GPU). Google Colab (Google Colaboratory) is a free Google cloud service that helps the AI research community develop deep learning applications by providing free GPUs and TPUs (we only need to sign up for an account. Google and use Google Colab in Google Drive).

Currently, Google Colab only supports GPU is Tesla K80 and TPU is TPUv2. The following table is a benchmark comparing CPU, GPU and TPU in Google Colab. Based on the comparison results, we see that TPU is almost twice as fast as the GPU and 10 times faster than that. CPU.

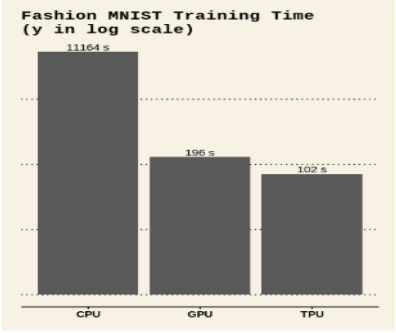

Fig. 2-3: Speed when training a model with CPU, GPU, TPU

From the above benefits that Google Colab brings, we have chosen Google Colab, the training model we deploy directly on Google Colab to make the model training process fast and efficient. and save more time.

2.3 Prepare the data

Our model trained 10 subjects including leaf curl, leaf spot disease, mildew, healthy leaf, human face, early chlorosis, ring spot mildew, bacterial spot disease, and two other diseases tomato tree. The main detection and identification diseases in the topic are speckled disease and morning blight, the remaining 8 subjects are used to enhance the ability to classify and help the model increase the ability to learn to avoid underfitting during digging. create. The table below is the visual statistics that we used in the topic, the subjects for erection were already images from the PlantVillage dataset, and the two main subjects were leaf spot disease and dew disease. The images are mostly collected online, some taken from the PlantVillage database for the wrong purpose of enhancing the training data set.

Table 1: Number of training images for the model

Training subjects	Number of photos
Healthy leaves	270
Many other diseases (6 types)	930
Leaf spot disease	570
Morning dew	540
Human face	300

It will then convert file.csv with the file containing .jpg image to tensorflow's format tfrecord.

After creating two files train.record and file test.record, we proceed to create the file labep_map.pbtxt. In this file contains the id of each object that I have labeled and in this file includes all the labels of the object I trained for the model.

2.4 Model training

To carry out the model training process, we need to prepare the model and deceive to choose the right model for the topic that has important factors to what we want when studying the problem of object detection and recognition. The table below shows the models provided by tensorflow for the user to use, including all the numbers that have been verified for average speed and accuracy.

Based on the table of Fig.s tensorflow provided, we chose the model ssd_mobilenet_v2_coco to serve our topic. In general, the ssd_mobilenet_v2_coco model has a relatively accurate speed, high actual running speed, so many people use it when deployed on computers with low configuration, or embedded computer versions such as Raspberry Pi.

In this file, the models have pre-existing training model, frozen_inference_graph file. pb. This file can detect many things like objects, people, animals, ... In order to build our own model, we need to conFig. the ssd_mobilenet_v2_coco.config file, which is provided by tensorflow inside the object detection file that is mistaken for users to easily access deep learning.

For convenience as well as to save time. We proceed to upload all the necessary files to Google Colab to conduct the model training process. Google Colab has a downside that every 12 hours of using the system will reset itself and easily lose data, so to avoid losing files after digging. We have submitted all the data required for the model training to Drive and then created the link from Google Colab to the drive.

3. CONCLUSION

3.1 3.1. Hardware after completion

Fig. 3-1: Finished hardware model

Compact design model using only V1 geared motors, RC servo motor, proximity sensor, H-bridge circuit and the microprocessor part is Raspberry Pi 3B +, aiming at sampling for processing. During the operation, the car works according to the algorithm we have programmed. The next model consists of a rail that uses plastic pipes to assemble, and a sample holder of the diseased leaf image (consisting of two diseases: spot and morning mist). The main task of autonomous vehicles is mainly going back and forth using a servo to take samples (tomato plant leaves).

3.2 Model evaluation results

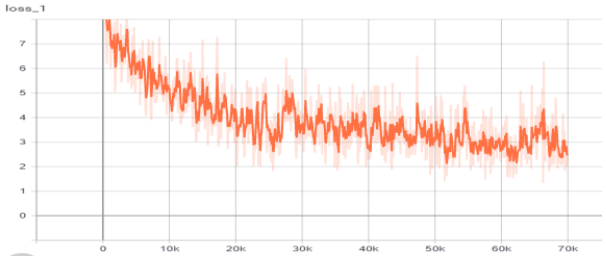

Fig. 3-2: Graph of loss function evaluating training process

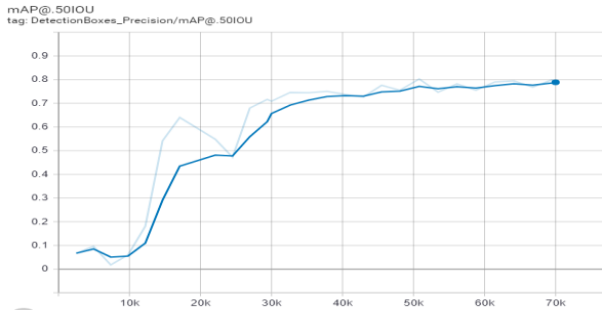

Fig. 3-3: Graph of IOU evaluating detection training

Based on the above two graphs, we found that the loss function decreased unevenly, unstable, the IOU function produced a relative match between the frame at training and the frame at labeling. The reason comes from the lack of sharp images, the objects labeled not vividly and the lack of image data for tomato diseases, leading to the loss of function stability.

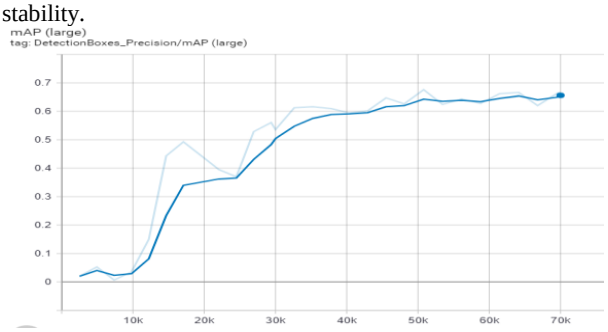

Fig. 3-4: Average accuracy of large object

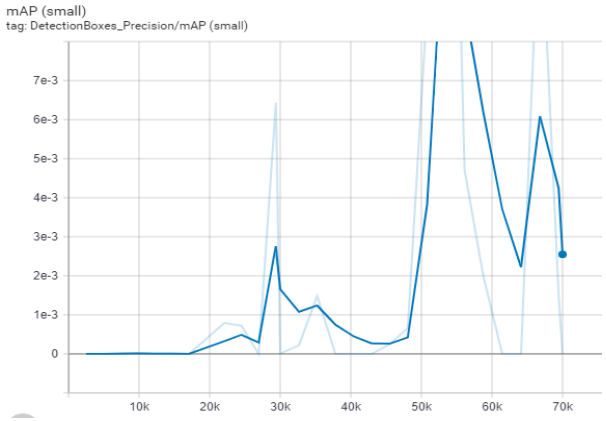

Fig. 3-5: Small object average accuracy

Based on the above two graphs, we can see that the average accuracy of the large image has a better probability, so it can be concluded that the ssd_moibenet_v2 model is only suitable for large objects.

3.3 Detection and identification results

Below are the results as we select online videos and images in the Plantvillage datasets published online to check the accuracy of our trained model.

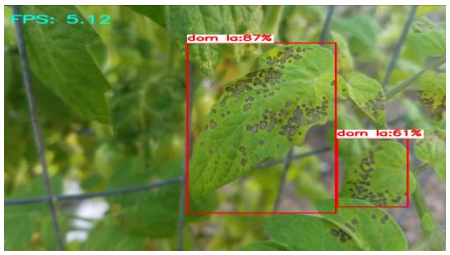

Fig. 3-6: Results of identification and detection of leaf spot disease

Fig. 3-7: Results of identification and detection of mildew (photo)

The two images above are the results of the detection and identification of tomato plant diseases. Data of 99%, 87%, and 67% of these numbers are percentages of probability to represent reliability when identifying the disease correctly. The two images above are just one of the tested images.

Fig. 3-8: Results of detecting leaf spot disease when experimenting

Fig. 3-9: Results of finding morning dew on experiment

The two pictures above are tested from the online image and bring out color photo for self-driving vehicle identification. These two plates are selected by us because they give a relatively high accuracy and good detection and identification capabilities, so we put them into the results.

The image below shows the results of the detection and identification of the tomato plant and was sent to the mail to alert the user. Attached mail includes image file and detected disease name.

Above are the visual results of the experiment, below is the table of data when we conduct the experiment. With data taken from the Plantvillage data set published by the researchers for a disease classification competition.

Table 2: Experimental examination of 100 apricot blossom samples and 100 leaf spot samples

Disease Sample	Sample Test	Correct finding	Average accuracy
Suong mai	100	94/100	92%
Leaf spots	100	85/100	84%

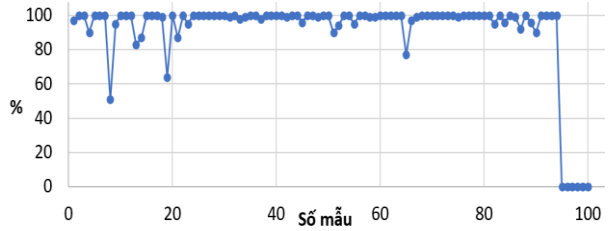

Fig. 3-10: Diagram for evaluating accuracy of 100 samples of morning mist

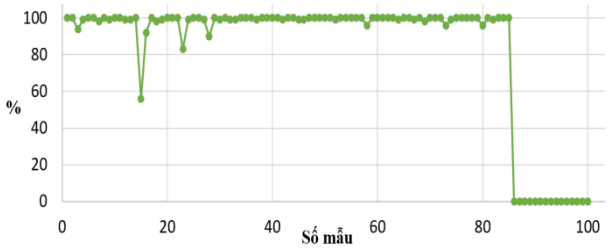

Fig. 3-11: Diagram for evaluating accuracy of 100 leaf spot disease samples

From the two tables of results and the graph of experimental data on 100 leaf spot disease samples and 100 mildew disease samples were taken from Plantvillage data set. Experimental results on this sample showed that the correct recognition of morning blight was 94/100 diseases (the rest gave two diseases and some detected wrong diseases), leaf spot disease was 85/100 diseases (the rest showed two diseases and some found false diseases), with relatively high recognition and classification accuracy when experimented with Plantvillage's image set. Based on the data as well as the accuracy, we can see that the ability to detect and identify diseases on leaves affected by leaf spot disease and morning blight gives relatively good results. However, when implementing the actual running model, there are still many limitations on the experimental model as well as other factors that affect the results of disease identification and detection, thus producing the detection results.

4. REFERENCES

[1] Q. H. Cap, K. Suwa, E. Fujita, S. Kagiwada, H. Uga, H. Iyatomi, An End-To-End Practical Plant Disease Diagnosis System For Wide-Angle Cucumber Images, International Journal of Engineering & Technology, 2018, 106-111.

[2] Marko Arsenovic, Mirjana Karanovic, Srdjan Sladojevic, Darko Stefanovic, Solving Current Limitations of Deep Learning Based Approaches for Plant Disease Detection, 2019.

[3] Amanda Ramcharan, Peter McCloskey, Kelsee Baranowski, Neema Mbilinyi, Latifa Mrisho, Mathias Ndalahwa, James Legg, David P. Hughes, A Mobile-Based Deep Learning Model for Cassava Disease Diagnosis, 2019.

[4] Lalitpur, Dhapakhel, TOMATO PLANT DISEASES DETECTION SYSTEM USING IMAGE PROCESSING, 2018.

[5] Alexander Driaba, Volgograd, Recognition of Various Objects from a Certain Categorical Set in Real Time Using Deep Convolutional Neural Networks, 2019.

[6] Manhnh, HOW TO SELECT RIGHT DEEP LEARNING MODEL FOR OBJECT DETECTION APPLICATIONS, 2019.

[7] Chengwei, How to train an object detection model easy for free, 2019.

[8] Https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_model_zoo.md

[9] Https://www.kaggle.com/emmarex/plantdisease