A SEQUENTIAL CONVOLUTIONAL NEURAL NETWORK FOR DETECTING TOMATO (*Solanum Lycopersicum* L.) FOLIAR DISEASES

A Computer Science Learning Evidence
Presented to the Faculty of the College of Information and Computing
University of Southeastern Philippines
Bo. Obrero, Davao City

Learning Evidence Submitted in Partial Fulfillment of the Requirements for
CS 3211 – CS RESEARCH METHODS

De Villena, Francis Nathanael
Moreno, Krisitan
Bordios, Kent Cyril

Adviser
Pilongo, Genevieve

April 2022

TABLE OF CONTENTS

List Of Figures

List Of Tables

1. INTRODUCTION

1.1 Background of the Study

The edible fruit of the cultivated tomato plant (*Solanum lycopersicum* L.) is the world's most substantially consumed vegetable attributable to its commodity as an essential ingredient in a sizable range of raw, cooked, or processed foods [1,2,3]. Moreover, it is a rich source of fiber, potassium, and vitamins A, C, and K, and also a dietary source of antioxidants. It contains carotenes, such as lycopene and beta-Carotene that give the fruit predominantly red and orange color, as well as tomatine, an alkaloid with fungicidal properties and its concentration determines the taxonomy of the species [1,2]. Tomato belongs to the genus Solanum within the large and diverse family Solanaceae, also known as nightshades, which includes several other commercially important species such as potato (*Solanum tuberosum* L.) and eggplant (*Solanum melongena* L.) [1,3]

Tomato plants are mostly branched and somewhat trailing when fruiting but some strains are compact and upright. Odorous and hairy glandular trichomes cover the plant's angular stem as well as its leaves that are varied in shape from lobed to compound, with pinnately arranged segments. Varieties with compound leaves consist of petiolate and dentate leaflets. It also has pendent, actinomorphic, five-petaled, yellow flowers blooming in clusters. Its fruits are globular or ovoid depending on varieties, still, they exhibit every characteristic of berries. The bilocular or multilocular cavities of each fruit contain small lens-shaped seeds enveloped by gelatinous pulp. The fruits have thin exocarp and their fleshy tissue comprises the remainder of the pericarp as well as the placenta. In addition, the fruit color is derived from the carotenoid content of the fruits' flesh [1].

Thanks to selective breeding of varieties, genetic engineering, and advancements in horticulture, the said species has wide climatic tolerance from temperate to tropical regions around the world and achieved cosmopolitan distribution in the present day [1,5]. Despite being one of the most valuable crops in the world, tomato cultivation is also highly labor-intensive. Not only does it require ample growing conditions to flourish but is also susceptible to numerous microbial pathogens and pests in its entire life cycle [1,2,3,5]. Typically, most of them would not kill the tomato plant, still, the damage to its roots, foliage, and fruit can cause a severe deficit in harvests and degradation of fruit quality [1,2]. Pythopathological diseases and disorders are primarily responsible for substantial yield losses and disruption of the global and local tomato supply. Therefore, the aforementioned problems are some of the reasons that commercial tomato production has shifted from open-field agriculture to protected horticulture [5]. The majority of traditional tomato cultivators rely heavily on chemical pesticides and fungicides to protect crops, yet, excessive and indiscriminate use of which leads to the possibility of resistance, increased production expenses, as well as adverse repercussions to human and environmental health [6].

Not to mention, pesticides and fungicides are futile against infectious microbial and viral diseases. Therefore, disease prevention and management strategies should also be taken into account, as these methods can drastically reduce the probability of infection and may mitigate the impact of an outbreak in case it befalls [7]. For instance, opting to grow cultigen tomatoes that manifest tolerance or resistance to specific pests and diseases can be practical when the circumstances are unmanageable, although the chosen variety should adapt to the farm location and produce a competitively marketable yield [1,2]. Another alternative method is the deployment of biological control agents to suppress the population of pests as well as pathogenic microorganisms and abate their disfigurement to crops, whilst generally harmless to the plant and surrounding environment since biocontrol agents only target their natural enemies. However, deployment and operational costs of biological control are expensive, and agents can never eradicate pest organisms otherwise they obliterate their food source [8]. There are other various proven and reliable treatment approaches intended for particular issues, but bear in mind that the effectiveness of these control measures depends on the accurate identification of diseases and the causal agents. Otherwise, misdiagnoses will squander financial resources and effort, in the worst case, it can also exacerbate yield losses [9,10].

For this reason, the first crucial step in taking action is to investigate the characteristic symptoms, then identify the disease causal agents or infesting pests, before deciding on which appropriate and cost-effective strategies to implement [9,10]. Early detection can also avert disease outbreaks and quell pests in time as well, even though it involves constant monitoring and inspection. Phytopathologists and specialists from related disciplines usually perform disease diagnosis and pest identification through on-site observation or isolating samples by conducting further tests in a laboratory, if the information gathered from the immediate examination is insufficient [9,10]. Considering the scale of the farm and the severity of the case, the whole diagnostic process may be time-consuming and costly [10].

With the tremendous growth of chip processing power and drastic improvement of machine learning algorithms, these advances have enabled the deep learning methods to effectively exploit complex, compositional nonlinear functions, learn distributed and hierarchical feature representations, and make effective use of both labeled and unlabeled data [11,12]. In turn, deep learning is extensively applied across various fields in both the scientific community and industry sectors and they have achieved efficacious performance. Augmenting deep learning concepts with image processing to customary diagnostic techniques in plant disease and pest classification will generate accurate results whether the crop is infected or not at instantaneous speeds [13-19].

1.2 Statement of the Problem

Some small-scale tomato growers, especially in the rural areas of the Philippines, might not afford to hire specialists and have no immediate access to assistance at the onset of disease in their crops; they would rather investigate the symptoms and attempt to treat the crops' ailments by themselves. Besides, if they lack proper knowledge of entomology and phytopathology of tomatoes, they can only draw subjective judgments based on experience and confront the problem through trial and error, which can be quite cumbersome and ineffective. When the treatment is handled erroneously, the disease could spread further and damages to the crops would worsen. Furthermore, in the case of lucrative profit-oriented tomato production in the country, the massive scale of farms made the continual monitoring of crops laborious, requiring more manpower as well as resources. This could be disadvantageous since early detection of diseases and quick response is crucial for reducing damages to crops. Also, commercial tomato growers place importance on minimizing operational expenses to earn more profit. Hence, the proponents observed the need for rapid and reliable diagnosis of insectile, microbial, and viral diseases manifested in the foliage of tomato plants, through deep learning.

1.3 Significance of the Study

This study aims to realistically apply deep learning, particularly the sequential convolutional neural network, in detecting tomato foliar diseases. The experimental findings are hoped to contribute to the development of practical systems or mobile applications that is focused on the diagnosis of tomato diseases, regardless of tomato cultivar and variety. This will benefit both small-scale and commercial tomato growers in the Philippines to improve their pest and disease management. In addition, the results of the proposed model's performance can also be used as a reference for other studies that are involved in designing a superior neural network for identifying diseases of other economically valuable plants. Lastly, the proponents intended to demonstrate the significance of deep learning to the agricultural sector for Philippine policymakers and agriculturists.

1.4 Scope and Limitations

The coverage of this study is only exclusive to the diseases that manifest on the foliage of a tomato plant. Diseases and disorders of the tomato's fruit, stem, and root are ruled out. Nevertheless, this study is limited to proposing a neural network model, and the external libraries of the Python programming language used for its implementation might not be available to other languages. Note that there will be no further development of any system or application.

1.5 Definition of Terms

Accuracy – a percentage of predicted values that matches the actual.

Adaptive Moment Estimation – also known as Adam; a stochastic gradient descent extension that iteratively updates network weights depending on training data.

Area Under the Curve – also known as AUC; the percentage of this area that is under this ROC curve, ranging between 0 to 1.

Batch Size – the number of samples processed through to the neural network at one time. Each sample in a batch is processed independently, in parallel with the other samples.

Categorical Cross-Entropy – a measure of the difference between two probability distributions for a given random variable or set of events applied to a multi-class classification task.

Convolution – a specialized linear operation on two functions of a real-valued parameter to generate a third function indicating how the shape of one function is transformed by another.

Convolutional Neural Network – a deep learning method that is comprised of many key components, namely, convolution layers, pooling layers, and fully connected layers, and is designed to automatically and adaptively learn spatial hierarchies of features using a backpropagation algorithm.

Deep Learning – a subset of machine learning techniques that use several layers of nonlinear information processing for supervised or unsupervised feature extraction and transformation, as well as pattern analysis and classification to learn several levels of representation and abstraction that aid in the interpretation of data such as images, sound, and text.

Dropout – randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting.

Epoch – an arbitrary cutoff in training, defined as one single pass over the entire training dataset to the network.

Foliage – the leafy parts of a plant.

Fully-connected Layer – a layer used in the last stages of a neural network to change the dimensionality of the output from the preceding layer so that the model can readily describe the relationship between the values of the data on which the model is working.

Gradient – the slope of a function and measures the change in all weights concerning the change in error.

Keras – Google's high-level deep learning API for creating neural networks. It is built in Python and is used to make neural network implementation simple. It also allows for the calculation of various backend neural networks.

Layer – the fundamental building elements of neural networks It is made up of a tensor-in tensor-out computing function and some state, which is stored in variables.

Learning Curve – a plot depicting the progression of a certain learning metric overtime during the training of a deep learning model, with a mathematical description of the learning process

Loss – a measurement of how far a model's predictions are from its label and represents how good or bad a model is.

One-Hot Encoding – a process in which categorical variables are converted into a suitable format for the model prediction.

Optimizer – also known as loss functions; used to minimize loss and adjust input weights, by comparing prediction and the loss function.

Pooling – a method for generalizing features retrieved by convolutional filters and assisting the network in recognizing features regardless of where they are in the image.

Receiver Operating Characteristic Curve – also known as ROC curve; graphic plot illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.

Rectified Linear Unit – also known as ReLU; a piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.

Sample – an element of a dataset.

Sequential Model – a linear stack of neural network layers ordered sequentially, with only one input and output tensor at each layer.

Shape – a list or tuple of numbers containing the size of each dimension of a tensor object.

Tensor – a generalization of vectors and matrices that are n-dimensional and contain the same type of data.

TensorFlow – Google's open-source library designed particularly for deep learning applications, offering a comprehensive machine learning platform with both high-level and low-level features for developing and deploying machine learning models.

2. REVIEW OF RELATED LITERATURE

2.1 Deep Learning

Deep learning is a subset of machine learning that learns to represent the world as a nested hierarchy of concepts, with each concept defined in terms of simpler concepts and more abstract representations calculated in terms of less abstract ones. It enables computer models built of several processing layers to learn data representations with varying levels of abstraction. Furthermore, deep learning identifies sophisticated structures in big data sets by employing the backpropagation algorithm to determine how a machine's internal parameters that are used to calculate the representation in each layer from the representation in the previous layer should be changed [11,12]. A deep-learning architecture is a multidimensional stack of basic modules, the majority of which can learn and many of which calculate non-linear input-output mappings. Each module in the stack alters its input to improve the representation's selectivity and invariance. A system with several non-linear layers can design exceedingly complicated functions of its inputs that are sensitive to minute details while being insensitive to significant irrelevant fluctuations. Many deep learning applications employ feedforward neural network topologies, which learn to convert a fixed-size input to a fixed-size output. To advance from one layer to the next, a group of units computes a weighted sum of their previous layer's inputs and passes the result via a non-linear function [11,12].

2.2 Convolutional Neural Network

Convolutional networks, also known as convolutional neural networks (CNNs or ConvNets), are a type of neural network that processes information using a predefined, matrix topology. In at least one of their layers, CNNs utilize a mathematical operation called convolution, hence the name, instead of generalized matrix multiplication. Convolution, in its most generic form, is conducted on the input data of a CNN using a filter or kernel, also known as a feature detector, to generate a feature map. CNNs are often applied for visual imagery, assisting a machine to recognize and learn from digital images. It maps the input image using a matrix of pixel values, and then it performs image recognition and classification based on the numbers it reads, recognizes patterns, and learns from them [12,20].

CNNs have three types of layers: convolutional layers, pooling layers, and full-connected layers. The top layer and essential building block of a ConvNet is the convolutional layer, and it is where the majority of computing occurs. Input data, a filter, and a feature map are all crucial parts of this layer. The feature detector is a two-dimensional array of fixed weights that represents a portion of the image, and its width and height define the scale of the receptive field. The filter is then applied to a portion of the input image, and a dot product between the input pixels and the filter is computed. This dot product is then passed into an array of outputs. Following that, the filter shifts by a stride, which is the amount of the step the convolution filter moves each time, and the procedure is repeated until the kernel has swept through the whole image. A feature map, activation map, or convolved feature is the ultimate result of a sequence of dot products from the input and the filter [12,20].

Padding is done to prevent the feature map from shrinking and having the same dimension as the input since the spatial extent of the feature map is always less than that of the input. Some feature detector parameters, such as weight values, are adjusted during training via backpropagation and gradient descent. CNNs have been able to drastically reduce the number of unique model parameters while also greatly increasing network sizes without requiring a comparable increase in training data owing to parameter sharing. However, four hyperparameters affect the volume size of the output, which must be specified before neural network training commences. These variables include kernel size, filter count, stride, and padding type. Ultimately, the convolutional layer encodes the image to numerical values, which the neural network may then analyze and extract significant patterns from [12,20]

Convolutional layers can pass their feature map with other convolutional layers or pooling layers. If another convolutional layer is added after the first, the structure of the CNN can become hierarchical since the subsequent layers can see the pixels inside the receptive fields of the preceding layers. Adding a pooling layer after the initial or subsequent convolutional layers, on the other hand, allows for dimensionality reduction, reducing the number of parameters in the input to decrease complexity, enhance efficiency, and lower the danger of overfitting. The pooling process, like the convolutional layer, sweeps a filter across the whole input, but this filter does not contain any weights. Instead, the kernel applies an aggregation function on the receptive field values, filling the output array. This layer aids in the reduction of complexity, the enhancement of efficiency, and the mitigation of the possibility of overfitting. It still employs feature mapping as the output, albeit a lot of information is lost when the output is sent to the next layer. [12,20]

The fully connected layer, the last layer in convolutional networks, precisely characterizes itself. Every neuron in the output layer is fully connected to a neuron in the preceding layer in the said layer, which is comparable to the principle of multilayer perceptron neural networks (MLPs). This is where the image is classified based on the characteristics retrieved from the previous layers and their various filters [12,20].

The CNN becomes more complex with each layer, recognizing more portions of the image. Earlier layers extract basic features like colors and edges. Deeper layers would then integrate these features, and final layers would recreate the entire image, beginning to identify bigger elements or forms of the object until it eventually recognizes the particular object. Because CNNs outperform other neural networks in image classification and object recognition, they have been applied in the identification of tomato leaf diseases, obtaining very accurate results as demonstrated by multiple studies [21-24] and might be adapted to identify other plants diseases.

2.3 TensorFlow-Keras Sequential Model

The Sequential model is the simplest and most common ConvNet model since it is a linear pipeline of neural network layers arranged sequentially, with only one input and output tensor at each layer. The Sequential model makes use of input interactions as well as non-linearities. The model comprises one input variable, two neurons in the hidden layer, and one binary output in the output layer. It sequentially transfers the data and fluxes from the first layer until the data reaches the end of the model. By stacking layers one after the other like building blocks, it is feasible to easily implement an easy-to-use, modifiable Sequential model architecture with Keras and TensorFlow. However, there are certain drawbacks to the model: it lacks flexibility because of its constrained topology; the model and its layers cannot contain many input and output tensors, and shared layers and branches are likewise not viable to configure [25].

2.4 Conceptual and Theoretical Framework of the Study

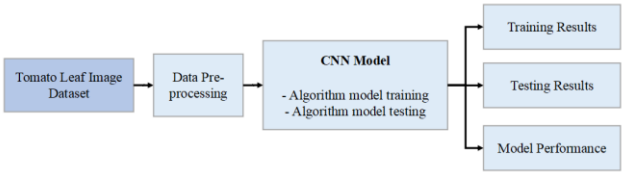

Figure 1. Research Conceptual and Theoretical Framework

The proponents will follow the structure of ConvNet Architecture for implementing the Sequential model. As shown in the figure above, the input will be the dataset comprising tomato foliage samples. Then, these images will then undergo pre-processing so the model will be able to process the data appropriately. After that is modeling, where we will define the model's layers and compile them. Once the model is created, we will begin the training process. The first half is feature extraction, that being the convolution and pooling layers. The other half would be classification, the output formed from the pooled feature maps will be converted into a single linear vector, this process is called flattening. Then, apply full connection and Softmax to the output. When the model is trained, we will execute evaluation and prediction to know the model's overall performance and representation of the dataset. If the model is precise in classifying the samples, then the study is concluded.

3. METHODOLOGY

3.1 General Methodological Approach

The researchers applied the Action Research design in this study to experiment, analyze and benchmark the performance of the algorithm in identifying the disease ravaging the foliage of the plant, specifically the tomato plant. The proponents utilized the Sequential ConvNet model with the help of Keras and TensorFlow to train, and benchmark the performance of the model in identifying the disease. The proponents will also utilize Jupyter Notebook in implementing the neural network model written mostly with Python. The data processing, modeling, training, and testing procedures were entirely performed in Google Colaboratory.

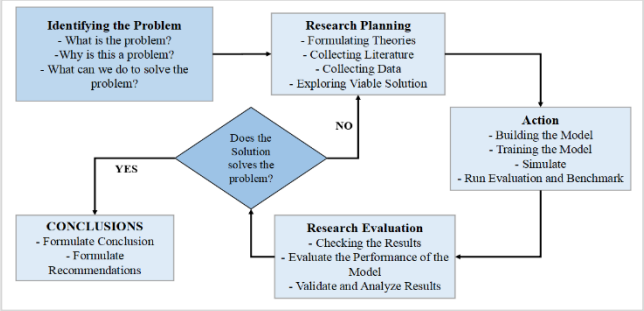

Figure 2. Action Research Design

The researchers utilize a very simple approach to Action Research design. Action Research design is the best research method to use in this study since the study aims to develop, evaluate and analyze a certain method to solve an existing problem. The first step the researchers considered is to identify the problem, every research design starts with identifying and understanding the problem to formulate or create a viable solution. Understanding the problem deeper also helps researchers on creating/formulate a certain specification for the solution. The next step after identifying the problem will be the Research Planning, this step involves: formulating theories, collecting literature for reference, collecting data, and exploring viable solutions for the problem. The Action step in the Research design involves building the chosen solution the researchers have deemed viable for the problem. This step involves the training of the model, simulation, evaluation of the model, and benchmarks. After the Action Step, the last step before going to conclusions is the Research Evaluation. The Research Evaluation steps involve checking the results and performance of the model and analyzing the results of if it's a viable solution to the problem. The final step is the conclusions after the model matches the specifications.

3.2 Data Collection

The proponents gathered samples from the PlantVillage dataset [26], retrieved from Sharada Mohanty's GitHub repository [27]. This particular dataset contains a total of 18,162 unique images of tomato leaves, divided into ten categories: two fungal diseases, two bacterial diseases, two Oomycete diseases, two viral diseases, a disease caused by a mite, and a healthy state of a tomato leaf.

Table 1. Overview of PlantVillage Dataset

Class	Category	Samples
0	Bacterial Spot	2027
1	Early Blight	1000
2	Late Blight	1909
3	Leaf Mold	952
4	Septoria Leaf Spot	1771
5	Two-spotted Spider Mites	1676
6	Target Spot	1404
7	Yellow Leaf Curl Virus	5357
8	Mosaic Virus	373
9	Healthy	1591

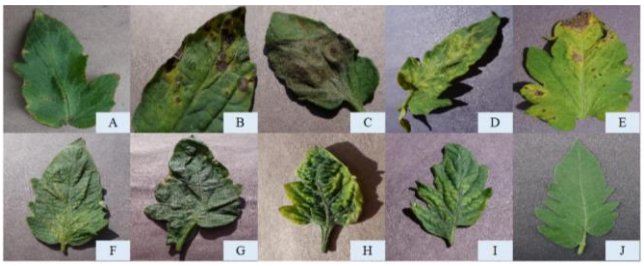

Figure 3. Samples of PlantVillage Dataset

The figure above shows a tomato leaf sample of every category from left to right: Bacterial Spot (*Xanthomonas perforans*), Early Blight (*Alternaria solani*), Late Blight (*Phytophthora infestans*), Leaf Mold (*Passalora fulva*), Septoria Leaf Spot (*Septoria lycopersici*), Two-spotted Spider Mites (*Tetranychus urticae*), Target Spot (*Corynespora cassiicola*), Yellow Leaf Curl Virus (*Begomovirus, Geminiviridae.*), Mosaic Virus (*Tobamovirus, Virgaviridae*), and the healthy one.

3.3 Data Processing

Data processing involves the preparation of the dataset for testing and training of the model. As the gathered dataset is already augmented, we can only rescale all sample images from 256×256 to 64×64 using the OpenCV library and distribute the samples into new different datasets for training and testing. The order contents of these datasets are randomly shuffled. Then, the data are reshaped for the data to be easily handled during the model training. The reshape function from the NumPy library is used in pre-processing the dataset for the model, it works similarly to vectorization, transforming a $m \times n$ matrix into a vector. In mathematics, particularly in linear algebra, vectorization refers to the linear transformation of matrices into a column vector. In the case of reshaping function: instead of transforming the matrix into a column vector, it is instead transformed into a row vector, removing the Transposition superscript. After that, the labels or class of the samples in training and testing datasets are converted to categorical one-hot encoding for model training and prediction.

3.4 Data Analysis

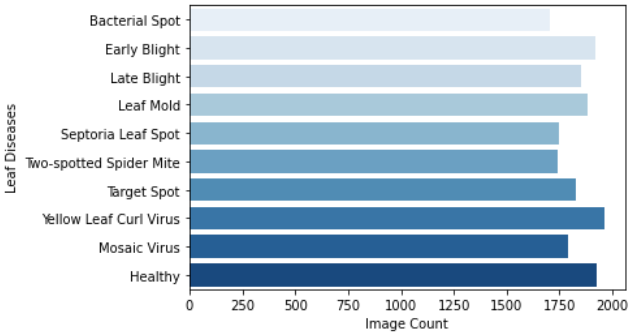

Figure 4. Overview of Training Dataset

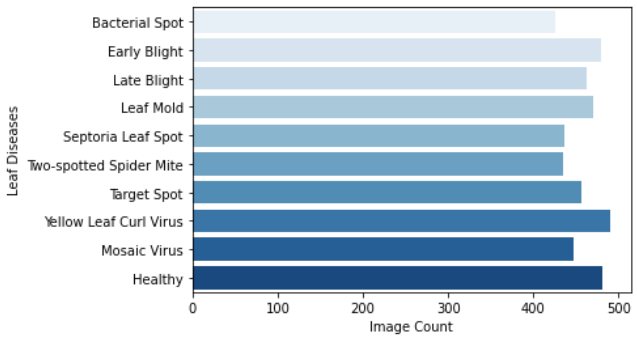

Figure 5. Overview of Testing Dataset

Figures 4 and 5 show the distribution of samples in every category. There are exactly 18,345 samples in the training dataset, while the testing dataset has 4,585 samples. The analysis of whether the training dataset is unrepresentative or not will be discussed in the next chapter.

3.5 Modeling

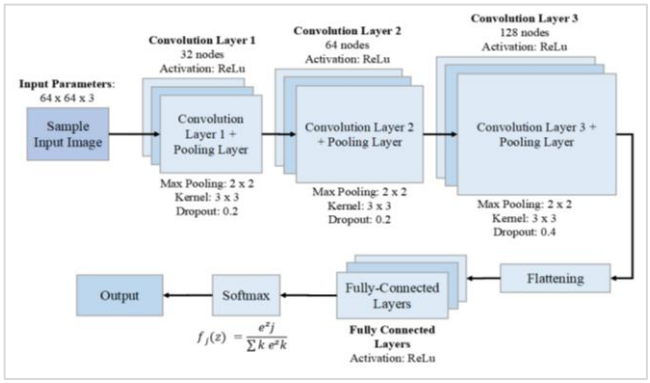

Figure 6. Architecture of the Proposed Sequential Model

The first thing to do when creating a sequential model is to initialize it and define its layers. Still, the model needs to receive information about its input shape, so we specified 64×64 input shape to the initial layer, which is the first convolutional layer. The model's subsequent layers can do automatic shape inference from the initial layer. Including the model's initial layer, we defined a total of three consecutive 2-dimensional convolution layers with the ReLU activation. Each of these layers has double the nodes from the previous one starting from the first with 32, 64, and 128 neurons, respectively. Succeeding after each convolution layer is an identical 2×2 max-pooling layer that reduces the resolution; thus, it optimizes the processing on subsequent layers. The dropout layer precedes every max-pooling layer, it randomly selects 20-40% of neurons to ignore during modeling training. This implies that on the forward pass, their contribution to the activation of downstream neurons is eliminated, and any weight changes are not transferred to the neuron on the backward pass. Essentially, dropout prevents overfitting. Next to the last dropout layer is the flattening layer. It transforms the feature map into a long 1-dimensional feature vector that will be then passed into the final phase of the model, the dense layers. We defined three successive dense or fully-connected layers with ReLU activation and 64 output shape, whereas the final fully-connected layer has a Softmax classifier function for activation and 10 output shape.

After defining the layers, the proponents compile the Sequential model before training to configure its learning process. Here, we specify some additional properties, namely loss function, and optimizer. For the loss function, we selected categorical cross-entropy since it is suitable for our study that involves multi-class classification and we used one-hot encoded labels for the model. We also chose the Adam algorithm with default values as the optimizer to train the model quickly and efficiently. Then, we added accuracy metrics to report the model's loss and accuracy during training as well as validation.

Once the model has been compiled, we trained the model for 16 epochs with 32 batch sizes. The tuples for fitting are the reshaped training dataset and one-hot encoded labels. Moreover, 25% of samples of the training dataset were split for validation. The results of the training process will be expounded in the following chapter.

Table 2. Summary of the Proposed Sequential Model

Layer	Output Size	Filter Size	Stride Size	Dropout	Parameter
Input Layer	64×64×3	---	---	---	---
Convolution	62×62×32	3×3	---	---	896
ReLU	62×62×32	---	---	---	---
Max Pooling	31×31×32	---	2×2	---	---
Dropout	31×31×32	---	---	0.2	---
Convolution	29×29×64	3×3	---	---	18,496
ReLU	29×29×64	---	---	---	---
Max Pooling	14×14×64	---	2×2	---	---
Dropout	14×14×64	---	---	0.2	---
Convolution	12×12×128	3×3	---	---	73,856
ReLU	12×12×128	---	---	---	---
Max Pooling	6×6×128	---	2×2	---	---
Dropout	6×6×128	---	---	0.4	---
Flatten	1×1×4608	---	---	---	---
Dense	1×1×4608	---	---	---	294,976
ReLU	1×1×4608	---	---	---	---
Dense	1×1×128	---	---	---	8,320
ReLU	1×1×128	---	---	---	---
Dense	1×1×64	---	---	---	8,256
ReLU	1×1×64	---	---	---	---
Dense	1×1×10	---	---	---	650
SoftMax	1×1×10	---	---	---	---
				Total Parameters	405,450
				Trainable Parameters	405,450
				Non-trainable Parameters	0

4. RESULTS AND DISCUSSION

In the course of the training process, the proponents assess model performance on both the training and validation dataset and plot the learning curve calculated from the accuracy and loss for each epoch. The shape and dynamics of a learning curve can be used to diagnose the learning behavior of a deep learning model, such as fitting speed. [28]. As shown in Figure 7, the curve of the training and validation loss has sharply declined to a point of stability and the gap between them is narrow. This implies that the Sequential Model improves fast as it is training in the beginning but slows down after a few epochs. The validation loss is slightly higher than the training loss as well which means the model has a fairly good fit on the validation dataset. Moreover, the training and validation

dataset is suitably representative since there is no significant noise on the learning curve of both training and validation loss. On the other hand, the training and validation accuracy curves rose sharply in the first few epochs and started to plateau later on, as seen in Figure 8.

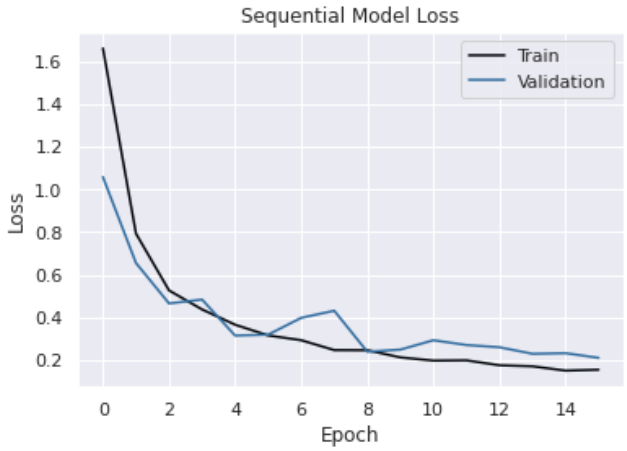

Figure 7. Optimization Learning Curve

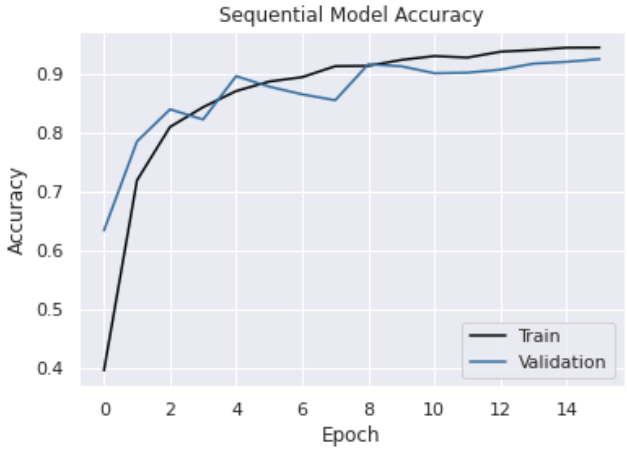

Figure 8. Performance Learning Curve

On the completion of model training, the model is then applied to the test dataset to check whether it is the best fit for diagnosing tomato foliage diseases and the corresponding test dataset. The results from the evaluation show that the model achieved a test loss of 19.86% and a test accuracy of 93.24%, therefore, it is viable to identify the diseases of tomato foliage.

Table 3. Evaluation Results

Testing Metric	Result
Loss	19.86%
Accuracy	93.24%

To further ensure the precision of the model, it performed a prediction of the disease of every sample in the test dataset. The Receiver Operator Characteristic (ROC) plot shown in Figure 9 illustrates that three among the ten given classifiers of foliage diseases had an area under the curve (AUC) of 1.000 while the rest are above 0.995, meaning these classifiers can perfectly distinguish between all the positive and the negative class points correctly.

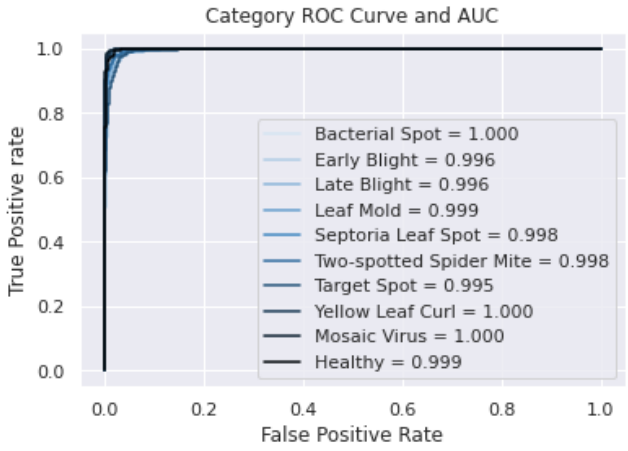

Figure 9. ROC-AUC

The confusion matrix shown in Figure 6 illustrates the efficiency and effectiveness of the classifier on a dataset for which the true values are given. True negatives (TN) occur when the class that was predicted to be negative turns out to be negative, such as not predicting a bacterial spot and the result is that it does not detect a bacterial spot. False negatives (FN) occur when the classifier predicts negative cases but it turns out to be positive, such as predicting no yellow leaf curve but it turns out to detect that said disease. The numbers shaded in blue are the true positives. False positives fall in the same row as the true positive. False negatives fall in the same column as the true positive, and true negatives do not belong in the same row or column as the true positive. The calculated accuracy of the classifier is ~0.988, meaning about 98.8% of the dataset was correctly predicted. Among the ten different conditions, the bacterial spot turned out to be the most accurate, followed by the yellow leaf curl virus. The least accurate condition was the early blight, wherein, its true positive is 0.86.

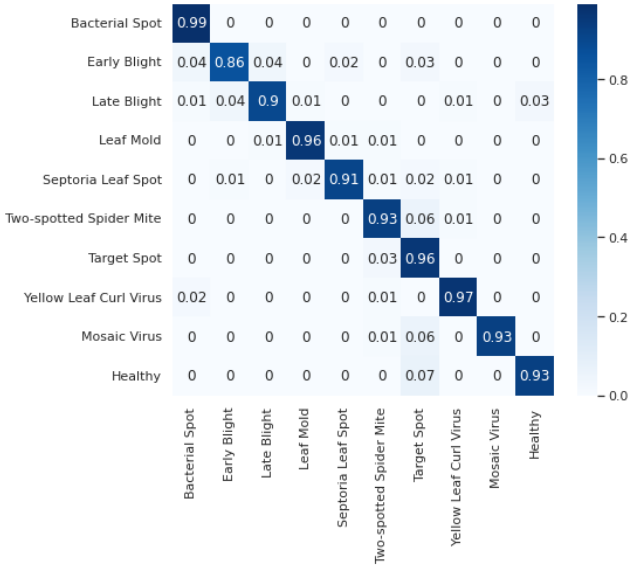

Figure 10. Confusion Matrix

5. SUMMARY. CONCLUSION, AND RECOMMENDATION

According to the results and evaluation of the model, the algorithm's performance in terms of accuracy and efficiency is highly satisfactory. The proposed model used in the study satisfied the specification and the performance evaluation. Therefore, the said deep learning model is a viable solution to the research problem.

In order to produce more accurate results, the image dataset in future researches has to be of high quality, as lower quality images will not help in the accuracy of the results. Deeper learning is needed not only to improve the accuracy, of the given conditions but also to discover more possible conditions out of the image dataset.

The farmers must also be well-equipped with proper agricultural knowledge about treating and preventing the spread of the disease as well as plant and crop production. Farmers who are relying only on their experiences without proper scientific knowledge in agriculture would have higher chances of failing or destroying their crops. Agricultural research and extension services are currently offered by state universities in the Philippines, encouraging farmers to attend seminars and training workshops such as handling diseased crops and disposal. Furthermore, additional research must be conducted on integrating the said algorithm into a mobile phone, as mobile phones are the most accessible piece of technology anyone can get their hands into.

APPENDICES

Sequential Model – Jupyter Notebook Source Code:

https://github.com/Frobbly/Python/blob/main/Tomato%20Foliar%20Diseases/tomato_foliar_diseases.ipynb

ACKNOWLEDGMENTS

First and foremost, we would like to give thanks to our most gracious and Almighty God who guided us and gave us strength in working on this project and the path of life. Secondly, our parents and relatives have always loved and supported us throughout our lives. Lastly, we would like to thank our course instructor, Mrs. Genevieve Pilongo, for imparting to us the knowledge of the fundamental lessons of Research Methods that is considered one of the most important cornerstones of our program, Computer Science, and for advising us throughout this project.

REFERENCES

[1] OECD. 2017. *Safety Assessment of Transgenic Organisms in the Environment, Vol. 1: OECD Consensus Documents*. Harmonization of Regulatory Oversight in Biotechnology, OECD Publishing, Paris, France. DOI: http://dx.doi.org/10.1787/9789264279728-en

[2] Trivedi, M., Singh, R., Shukla, M., and Tiwari, R. 2016. GMO and Food Security. In *Eco-friendly Pest Management for Food Security*, 703-726. DOI: https://doi.org/10.1016/B978-0-12-803265-7.00023-3

[3] Kelley, W., Boyhan, G., Harrison, K., Summer, P., Langston, D. Jr., Sparks, A. Jr., Culpepper, S., Hurst, W., and Fonsah, E. 2017. *Commercial Tomato Production Handbook*. University of Georgia, Athens, GA, USA.

[4] Baranski, R., Goldman, I., Nothnagel, T., and Scott, J. 2016. Improving Color Sources by Plant Breeding and Cultivation. In *Handbook on Natural Pigments in Food and Beverages: Industrial Applications for Improving Food Color*, 429-472. DOI: https://doi.org/10.1016/B978-0-08-100371-8.00019-1

[5] Hanssen, I., and Lapidot, M. 2012. Major Tomato Viruses in the Mediterranean Basin. *Advances in Virus Research*, 84 (2012), 31-66. DOI: https://doi.org/10.1016/B978-0-12-394314-9.00002-6

[6] Ramasamy, S., and Ravishankar, M. 2018. Integrated Pest Management Strategies for Tomato Under Protected Structures. In *Sustainable Management of Arthropod Pests of Tomato*, 313-322. DOI: https://doi.org/10.1016/B978-0-12-802441-6.00015-2

[7] Gajanana, T., Moorthy, P., Anupama, H., Raghunatha, R., and Kumar, G. 2016. Integrated Pest and Disease Management in Tomato: An Economic Analysis. *Agricultural Economics Research Review*, 19, 2 (2006). 269-280. DOI: https://doi.org/10.22004/ag.econ.57763

[8] Holmes, L., Manjiny, S., and Upadhyay, D. 2016. Biological Control of Agriculture Insect Pests. *European Scientific Journal*, 12, 10 (2016). DOI: https://doi.org/10.19044/ESJ.2016.V12N10P%25P

[9] Melanson, R. 2017. *Common Diseases of Tomatoes*. Mississippi State University Extension Service Publication 3175. Mississippi State University, Starkville, MS, USA.

[10] Shankar, R., Harsha, S., and Bhandary, R. 2014. *A Practical Guide to Identification and Control of Tomato Diseases*. Tropica Seeds Pvt Ltd, Bangalore, India.

[11] Goodfellow, I., Bengio, Y., and Courville, A. 2016. *Deep Learning*, MIT Press, Massachusetts Institute of Technology, Cambridge, MA, USA

[12] LeCun, Y., Bengio, Y., and Hinton, G. 2015. Deep learning. *Nature*, 521 (2015), 436-444. DOI: https://doi.org/10.1038/nature14539

[13] Zhang, K., Wu, Q., Liu, A., and Meng, X. 2018. Can Deep Learning Identify Tomato Leaf Disease? *Advances in Multimedia*, (2018). DOI: https://doi.org/10.1155/2018/6710865

[14] Brahimi, M., Boukhalifa, K., and Moussaoni, A. 2017. Deep Learning for Tomato Diseases: Classification and Symptoms Visualization. *Applied Artificial Intelligence*, 31, 4 (2017), 1–17. DOI: https://doi.org/10.1080/08839514.2017.1315516

[15] Mohanty, S., Hughes, D., and Salathé, M. 2016. Using Deep Learning for Image-Based Plant Disease Detection. *Frontiers in Plant Science*, 7 (2016), 1419. DOI: https://doi.org/10.3389/fpls.2016.01419

[16] Khurana, J., Sharma, A., Chhabra, H., and Nijhawan, R. 2019. An Integrated Deep Learning Framework of Tomato Leaf Disease Detection. *International Journal of Innovative Technology and Exploring Engineering*, 8, 11S (2019), 46-50. DOI: https://doi.org/10.35940/ijitee.K1010.09811S19

[17] Saranya, S., Rajalaxmi, R., Prabavathi, R., Suganya, T., Mohanapriya, S., and Tamilselvi, K. 2021. Deep Learning Techniques in Tomato Plant – A Review. *Journal of Physics: Conference Series*, 1767 (2021). DOI: https://doi.org/10.1088/1742-6596/1767/1/012010

[18] Atabay, H. 2017. Deep Residual Learning for Tomato Plant Leaf Disease Identification. *Journal of Theoretical and Applied Information Technology*, 95, 24 (2017), 6800-6808.

[19] Ashqar, B., and Abu-Naser, S. 2018. Image-Based Tomato Leaves Diseases Detection Using Deep Learning. *International Journal of Academic Engineering Research*, 2, 12 (2018), 10-16.

[20] Wu, J. 2017. *Introduction to Convolutional Neural Networks*. Nanjing University, Nanjing, Jiangsu, China.

[21] Rinu, R., and Manjula, S. 2021. Plant Disease Detection and Classification using CNN. *International Journal of Recent Technology and Engineering*, 10, 3 (2021). 152-156. DOI: https://doi.org/10.35940/ijrte.C6458.0910321

[22] Agarwal, M., Singh, A., Arjaria, S., Sinha, A., and Gupta, S. 2020. ToLeD: Tomato Leaf Disease Detection using Convolution Neural Network. *Procedia Computer Science*, 167 (2020), 293–301. DOI: https://doi.org/10.1016/j.procs.2020.03.225

[23] Zhao, S., Peng, Y., Liu, J., and Wu, S. 2021. Tomato Leaf Disease Diagnosis Based on Improved Convolution Neural Network by Attention Module. *Agriculture*, 11 (2021), 651. DOI: https://doi.org/10.3390/agriculture11070651

[24] Wu, Y., Xu, L., and Goodman, E. 2021. Tomato Leaf Disease Identification and Detection Based on Deep Convolutional. *Intelligent Automation & Soft Computing*, 28, 2 (2021), 561–576. DOI: https://doi.org/10.32604/iasc.2021.016415

[25] Chollet, C. 2020. The Sequential Model. Retrieved from https://keras.io/guides/sequential_model/

[26] Hughes, D., and Salathé, M. 2015. An open access repository of images on plant health to enable the development of mobile disease diagnostics. arXiv:1511.08060. Retrieved from https://arxiv.org/abs/1511.08060

[27] Mohanty, S. PlantVillage Dataset. Github. Retrieved from https://github.com/spMohanty/PlantVillage-Dataset

[28] Viering, T. and Loog, M. 2021. The Shape of Learning Curves: a Review. arXiv.2103.10948. Retrieved from https://arxiv.org/abs/2103.10948