PEST DETECTION ON TEA LEAF USING CONVOLUTIONAL NEURAL NETWORKS (CNN) METHOD

Bagja Hidayat[1], Galih Hermawan[2]

[1,2] Engineering Informatics – Computer University of Indonesia
Jl. Dipatiukur No 112-116 Bandung, 40132
E-mail : bagjahidayat48@gmail.com[1], galih.hermawan@email.unikom.ac.id[2]

ABSTRACT

Production and development of commodities of tea is one of the core business of the plantation PTPN VIII. One of the causes of the decline in tea production in Indonesia is the pest attack, there are two types of pests, namely *Blister blight* and pest *Hellopeltis*. In this study can be obtained through the four stages, namely, the first stage of the preprocessing method using grayscale, edge detection (Sobel) and Standardization, second-stage feedforward method using *convolutional layer*, *ReLU*, *max pooling* and *fully connected layer*, the third stage of backpropagation and the fourth stage of the confusion matrix. Convolutional Neural Network (CNN) is a method of classification to be used in this research, CNN is a variation of the Multilayer Perceptron (MLP) which inspiration of neural network humans. The objectives to be achieved, namely, convolutional neural network can detect the good in recognizing the types of pests in tea leaves. After testing the system with the Black Box method can be drawn the conclusion that the program works fine. In the process of testing data from the introduction that had been prepared a program capable of detecting the type of pest in tea leaves with an average accuracy of 95%.

Keyword: detection type pest, convolutional neural network, pattern recognition, tea leaf, multilayer perceptron

1. INTRODUCTION

Pests in the tea plant is a bully or a problem that isvery disturbing for the plantations. This pest can disrupt the growth of plants all of which are cultivated, including on the leaves.

As for the research of a type that is classification of type mango tree gadung and curut based on the texture of the leaf [1], *Classification Of Diseased Plant Leaves Using Neural Network* Algorithms [2], *Classification and Recognition of Herbal Leaf Using SVM Algorithm* [3] and the Detection of Defects the Tea Leaves of Camellia Sinensis with Digital Image Processing and JST *Learning Vector Quantization* [4].

From previous research has not been the presence detection type of pest in tea leaves using convolutional neural network methods. CNN is a type of neural network regular is used on the data image. CNN may be used to detect and identify objects in an image. Outline of CNN is not much different from neural network normally. CNN is composed of neurons that have weight, can and activation function. CNN has a special coating which is named with a layer of convolution where on this layer an image insert will be processed based on the filters that are already specified. Of each layer it will produce a pattern of some parts of the image that will be easier to be classified. This technique can make the learning function of the image become more efficient to implement.

Based on the above description in this study to identify an object where the method of convolutional neural network suitable for recognizing the types of pests in tea leaves so that in this final task model of CNN will be used as the detection of different types of pests on the leaves tea.

2. RESEARCH

2.1 Pattern recognition

Pattern recognition, namely politically symbolic and numeric data (including images) automatically by computer [5]. The goal of group is to recognize an object in the image. The computer receiving the form enter the image object that will identification the image, process and deliver outputs in the form of descriptions of objects in pattern recognition. Figure 1 shows the pattern recognition.

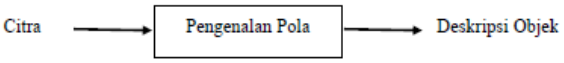


Figure 1. Pattern Recognition

2.2 Analysis of data input

The input data used is leaves image data taken from tea garden is located in Malabar Pangalengan Bandung District owned by PT. Nusantara Plantation VIII. Data input in the form of an image of leaf of developing pests, namely *Blister blight* and pest *Hellopeltis*. The number of image data, image data 400 440 made data training divided into 200 image of a leaf *blister blight* and pests developing 200 image leaves afflicted by pest *hellopeltis*, while 40 image data made the testing data is divided into 20 image leaf pests *blister blight* and leaf pest image 20 *hellopeltis*. The size of a pixel in this study i.e., 28x28

pixels. Example image of the leaves are stricken by the pests can be seen in Figure 2 and Figure 3.

a. Blister blight



Figure 2. Image of Leaf Pests *Blister blight*

b. Hellopeltis



Figure 3. The Image of A Leaf Pest *Hellopeltis*

2.3 *Preprocessing*

At this stage the image data RGB image processed from enter into grayscale, then changed to edge detection (sobel) and standardization. The output of this process is to be used on later stage i.e. feedforward. The stages of this process can be seen in Figure 4.

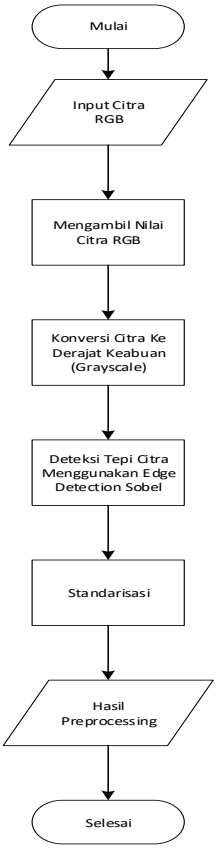


Figure 4. Flowchart Preprocessing

2.3.1 *Grayscale*

From RGB to grayscale, pixel values from samples taken 28x28 8 x 8 can be seen in table 1.

Table 1. The Value Of The Pixel RGB Image

xy	1	2	3	...	8
1	R = 93 G = 131 B = 39	R = 97 G = 135 B = 41	R = 92 G = 121 B = 26	...	R = 94 G = 136 B = 25
2	R = 102 G = 139 B = 51	R = 108 G = 141 B = 54	R = 103 G = 119 B = 35	...	R = 101 G = 139 B = 27
3	R = 112 G = 148 B = 61	R = 116 G = 151 B = 66	R = 100 G = 137 B = 44	...	R = 108 G = 143 B = 29
...	...	...	...	...	...
8	R = 103 G = 140 B = 46	R = 105 G = 141 B = 44	R = 91 G = 131 B = 23	...	R = 105 G = 143 B = 32

Example peritungan we take pixel (1.1) by the following formula.

(x, y) = 0.299 R + 0.587 G + 0.114.. B (1)

Then it can be illustrated as follows:

(1.1)= (0,2989 * R) + (0,5870 * G) + (0,1141 * B)
= (0,2989 * 93) + (0,5870 * 131) + (0,1141 * 39)
= 27.7977 + 76.897 + 4.4499
= 109.1446 ≈ 109.14

Then the overall grayscale can be seen in table 2.

Table 2. Results of Grayscale Pixel Values

xy	1	2	3	4	5	6	7	8
1	109,14	112,92	101,49	109,57	128,90	132,73	118,86	110,78
2	117,90	121,21	104,63	99,93	124,73	130,56	115,29	114,66
3	127,31	130,84	115,33	99,35	111,99	122,64	112,79	119,53
4	132,37	132,67	123,64	106,58	102,01	114,02	121,76	115,65
5	135,89	133,25	117,69	111,41	103,60	111,75	123,36	103,32
6	135,14	132,32	114,05	105,84	106,38	117,77	111,76	103,32
7	125,08	125,57	109,18	103,38	112,29	117,18	108,34	113,85
8	118,22	119,17	106,72	106,16	119,18	102,34	100,84	118,98

2.3.2 *Edge detection* (*Sobel*)

From the results of grayscale then in this stage was changed to grayscale in edge detection (Sobel) by the following formula Kernel Sobel

$$:x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad (2)$$

Formula:

$$x = ((1,3) + 2(2,3) + (3,3)) - ((1,1) + 2(2,1) + (3,1)) \qquad (3)$$

$$y = \big((3,1) + 2(3,2) + (3,3)\big) - ((1,1) + 2(1,2) + (1,3)) \qquad (4)$$

Gradient:

$$G = \sqrt{x^2 + y^2} \qquad (5)$$

Description:

G = Sobel Gradient

Example:

x = (127,3129+2*130,8400+115,3294) - (109,1446+2*112,9164+101,4924) = 67,8525

y = (101,4924+2*104,6332+115,3294) - (109,1446+2*117,8999+127,3129) = -46,1691

$G = \sqrt{67{,}8525^2 + -46{,}1691^2} = 82{,}0704 \approx \mathbf{82{,}07}$

Kernel sobel will be moved across the surface of stride one pixel grayscale resulting matrix of edge detection. The overall results can be seen in table 3.

Table 3. The Results of The Pixel Edge Detection

xy	1	2	3	4	5	6
1	82,07	85,09	68,43	107,70	67,54	58,03
2	72,53	123,79	12,61	101,13	50,75	20,49
3	50,19	107,24	63,32	41,57	63,18	10,36
4	66,66	98,38	57,87	23,25	64,68	41,34
5	83,81	102,39	30,74	40,79	26,95	43,08
6	81,87	88,26	12,60	37,23	35,73	22,12

2.3.3 *Standardization*

The process for changing or standard level of light intensity in the image, the image is mapped on a pixel with a certain size so as to provide a fixed dimension representation. Then it can be illustrated by the following formula.

$$s_j = \frac{x_j - \bar{x}}{\sigma} \quad (6)$$

Where:
xj = data to-j
x = the mean median
σ = standard deviation
Average:

$$\bar{x} = \frac{(1,1)+(2,1)+(3,1)+\cdots+(n,n)}{n} \quad (7)$$

Formula to calculate the standard deviation as follows:

$$\sigma = \sqrt{\frac{\sum_{j=1}^{n}(x_j - \bar{x})^2}{n-1}} \quad (8)$$

Description:
σ : standard deviation
xj : value x to-j
x̄ : average
n : sample size

1. Find the average value.

$$\bar{x} = \frac{82{,}07+85{,}09+68{,}43+107{,}70+67{,}54+\cdots+22{,}12}{36}$$

$$= \frac{2143{,}7821}{36} = 59{,}5495 \approx 59{,}55$$

2. Find the value of the standard deviation.

$$s_1 = \frac{82{,}0704 - 59{,}5495}{30{,}5921} = \mathbf{0{,}7362}$$

Then the results of the overall standardization can be seen in table 4.

Table 4. The Results of The Pixel Standardization

xy	1	2	3	4	5	6
1	0,74	0,83	0,29	1,57	0,26	-0,05
2	0,42	2,10	-1,53	1,36	-0,29	-1,28
3	-0,31	1,56	0,12	-0,59	0,12	-1,61
4	0,23	1,27	-0,05	-1,19	0,17	-0,60
5	0,79	1,40	-0,94	-0,61	-1,07	-0,54
6	0,73	0,94	-1,53	-0,73	-0,78	-1,22



The result of preprocessing stage can be seen in Figure 5.

Figure 5. Results of Preprocessing

2.4 *Convolutional Neural Network*

CNN is a variation of multilayer perceptron (MLP) inspired from human neural networks. Initial research underlying the invention was first by Hubel and Wiesel [6]. Convolutional neural network is a layer that has the order of neurons 3D (width, height, depth). Width and height is the measurement of the layer, while the depth refers to the number of layers. This CNN is a method to transform the original image layer to the layer of the image pixel value into the value of the score class for the classification. The analysis used in implementations of convolutional neural network can be seen in Figure 6.

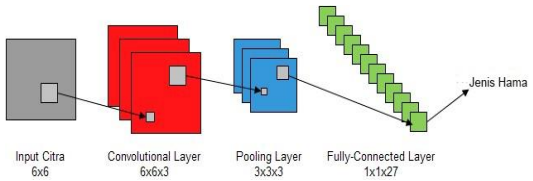


Figure 6. Convolutional Neural Network Analysis

2.5 CNN Network Architecture

At this stage of the process will be done from the beginning of input passed CNN convolution layer, activation function (ReLU), pooling and fully connected to layer produced a one-vector type classification hot pests.

2.5.1 *Convolution layer*

On a convolution operation will be done convolution layer between matrix of the input image with a kernel matrix. This kernel will be moved across the surface of the picture. The purpose of doing convolution on transformation the image was to extract features from the input image. Convolution will generate a linear from the input data according to spatial data information. Weights in the kernel convolution, specifies the layer that is used, so that the kernel can be trained convolution of the input based on CNN [8].

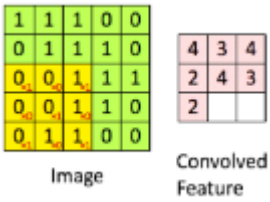


Figure 7. Convolution Operation

2.5.2 Function Activation *ReLU*

The activation function is a nonlinear function which allows a JST to be able to transform input data into higher dimensions so that it can be done, allowing simple hyper line cuts done classification. In CNN there is activation function is used i.e. ReLU functions.

Function activation ReLU, where the outcome of each pixel that has the convolution value will be 0 $0 <$ and that has a value of ≥ 0 the value set, so that the function activation ReLU has a form of the equation as follows:

$$f(x) = max(0, x) \quad (9)$$

2.5.3 *Pooling (Max pooling)*

Max output divides in relu pooling into several small grid and then take the maximum value of each grid to draw up a matrix of the image has been reduced as shown in Figure 8. The grid is red, green, yellow and blue is a cohort of the grid that will be the maximum value selected. So the result of this process can be seen in the collection of the grid on the right. These processes ensure features are obtained will be identical even though the image of the object undergoing translation (shift).

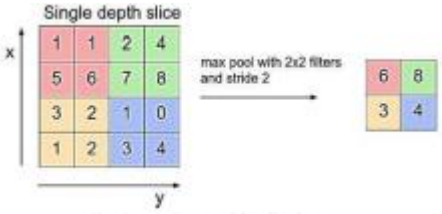


Figure 8. Operation Max Pooling

2.5.4 *Fully Connected Layer*

The layer is the layer that is typically used in the application of MLP and aims to perform a transformation on the data dimension so that the data can be classified in linear.

Each neuron in the convolution layer needs to be transformation into a single dimension data first before it can be incorporated into a fully connected can only be implemented at the end of the network. Figure 9 is a fully connected architecture layer used in this analysis.

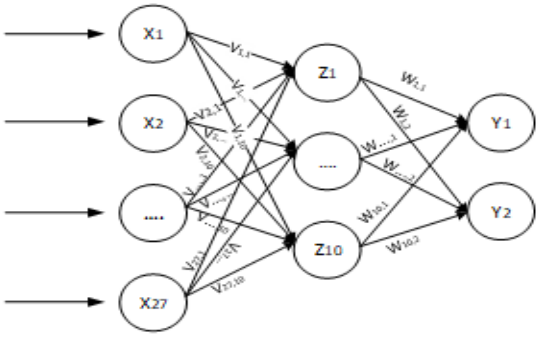


Figure 9. Fully Connected Architecture Layer

The resulting output is a vector one-hot type of vermin, money contains pest type insert image [7]. Karen kind of pest who used two (blister blight and hellopeltis), then the vector one-hot used has two components, i.e. y_1 and y_2 y_1, where this type of blister blight pest and pest *hellopeltis* type y_2.

2.6 *Process Training*

Training process is the stage where CNN is trained to obtain high accuracy of the classification is done. This stage consists of a process and the process of feedforward backpropagation. To begin the process of feedforward required number and size of the layer that will be set up, where the vector image is going through the process of convolution and Max pooling to reduce the size of the image and reproduce neuron. So many networks are formed where the Add variant data to study [8].

The result of the process of feedforward form lighter weight will be used to evaluate the process of neural network. Process flow as in Figure 10.

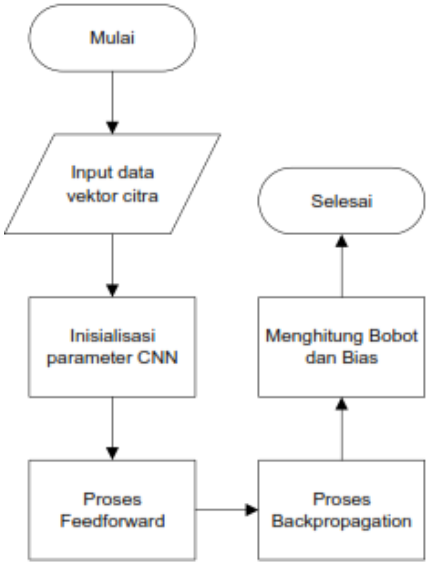


Figure 10. Process Flow Training

2.6.1 Process *Feedforward*

Feedforward process is the first stage in the process of training. This process will generate multiple layers of data for classifying image which uses weights and biases that have been updated from the process of backpropagation. This stage will also be reused when the process of testing.

2.6.2 Process *Backpropagation*

The process of backpropagation is the second stage of the process of training. The results of the process of feedforward on trace faults from the first layer to output layer. To indicate that the data has been obtained to trace the new weights and biases.

2.6.3 Gradient Calculation

The gradient for the convolution network is a process for obtaining a new weights and biases will be necessary when training and testing process.

2.7 Process Testing

The process of testing is a process for obtaining the value of weights and bias from the results of the process of training. This process is not much different from the process of training that sets it apart there is no backpropagation process after the process of feedforward. So the end result of this process of generating the accuracy of classification is done, the data that failed are classified, the number images that fail are classified and the shape of the network formed from the process of feedforward.

With weights and bias of the new process then generate a feedforward applied layers of output. The output layer are already fully connected with the provided label. The results are fully connected retrieved data that failed and successfully classified. From the explanation above of the form process flow testing as in Figure 11.

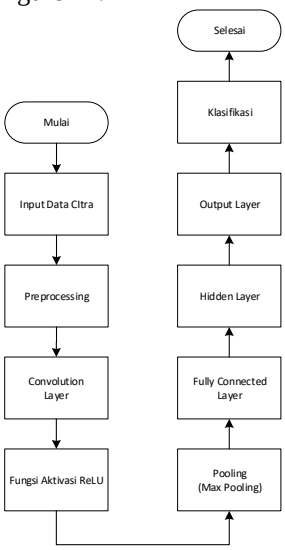


Figure 11. Process Flow Testing

2.8 Testing

Testing system is part of a search in order to find the fault in the system being built. The purpose of the testing system is to find a flaw in the system that was built before the system is used by the user. The done system testing includes functionality testing phase, testing detail design and performance testing.

2.8.1 Performance Testing

Performance testing is a phase that has the purpose to know the performance or accuracy of the results of the methods of Convolutional Neural Network (CNN) and how to calculate the amount of data a training class predicted correctly. How to measure the performance of the system will be evaluated using the confusion matrix.

Confusion matrix is a matrix that contains the actual data and the prediction data have been classified by the system [9]. In this study, performance testing is done using 1 methods with judgments can be seen in table 5.

Table 5. Testing Provisions Of The Confusion Matrix

Actual	Predict	
	0	1
0	TP	FN
1	FP	TN

Description:
0 = Positive
1 = negative

TP and TN give information when the correct classification, while the FP and FN tell when classification is wrong.

2.8.2 Test the input data that is not included in the training data

This testing is done with test data that is not included in the study data, the test aims to find out the level of introduction of data introduction to the learning data has been selected. Learning data is used as many as 400 data hellopeltis data, 200 and 200 blister blight data while the data is the introduction of as many as 40 20 data hellopeltis data and data 20 blister blight. For leraning rate set 0.01 and momentum 0.9 with epoch. Terms of performance testing data introduction can be seen in table 6.

Table 6. Provision Of Data Performance Testing

No	Nama File	Prediksi	Fakta	Status
1	blister_1.jpg	0 / *Positive*	0 / *Positive*	TP
2	blister_2.jpg	1 / *Negative*	0 / *Positive*	FN
3	hellopeltis_1.jpg	0 / *Positive*	1 / *Negative*	FP
4	hellopeltis_2.jpg	1 / *Negative*	1 / *Negative*	TN

2.8.2.1 Performance Testing of the First Testing Data

On this test using 400 data 40 data training and testing with the epoch = 5. As for performance testing of the first testing data can be seen in table 7.

Table 7. Confusion Matrix The First Testing Data

Predict	*Actual*	
	0	1
0	TP = 9	FP = 0
1	FN = 11	TN = 20

From table 7 confusion matrix the first testing data above can then be also measured the level of accuracy of predictions by using the following equation.

$$Accuracy = \frac{coreectly\ predict\ data}{total\ testing\ data} \times 100\% \quad (9)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \quad (10)$$

$$Accuracy = \frac{9+20}{9+20+0+11} \times 100\%$$

$$= \frac{29}{40} \times 100\% = 75\ \%$$

2.8.2.2 Performance Testing of the Second Testing Data

On this test using 400 data 40 data training and testing with the epoch = 10. As for testing the performance of the second testing data can be seen in table 8.

Tabel 8. Confusion Matrix First Testing Data

Predict	*Actual*	
	0	1
0	TP = 0	FP = 0
1	FN = 20	TN = 20

From table 8 confusion matrix the first testing data above can then be also measured the level of accuracy of predictions by using the following equation.

$$Accuracy = \frac{coreectly\ predict\ data}{total\ testing\ data} \times 100\%$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\%$$

$$Accuracy = \frac{0+20}{0+20+0+20} \times 100\%$$

$$= \frac{20}{40} \times 100\% = 50\ \%$$

2.8.2.3 Performance Testing Data to Third

On this test using 400 data 40 data training and testing with the epoch = 15. As for testing the performance of the second testing data can be seen in table 9.

Table 9. Confusion Matrix Testing Data to third

Predict	*Actual*	
	0	1
0	TP = 18	FP = 6
1	FN = 2	TN = 14

From table 9 confusion matrix the first testing data above can then be also measured the level of accuracy of predictions by using the following equation

$$Accuracy = \frac{coreectly\ predict\ data}{total\ testing\ data} \times 100\%$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\%$$

$$Accuracy = \frac{18+14}{18+14+6+2} \times 100\%$$

$$= \frac{32}{40} \times 100\% = 80\ \%$$

2.8.2.4 Performance Testing Data to Fourth

On this test using 400 data 40 data training and testing with the epoch = 20. As for testing the performance of the second testing data can be seen in table 10.

Table 10. Confusion Matrix Fourth Testing Data

Predict	*Actual*	
	0	1
0	TP = 20	FP = 2
1	FN = 0	TN = 18

From table 10 confusion matrix the first testing data above can then be also measured the level of accuracy of predictions by using the following equation.

$$Accuracy = \frac{coreectly\ predict\ data}{total\ testing\ data} \times 100\%$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\%$$

$$Accuracy = \frac{20+18}{20+18+2+0} \times 100\%$$

$$= \frac{38}{40} \times 100\% = 95\ \%$$

2.8.3 Testing Simulator

The following is a test that is performed against the data testing. Testing is done to see success and accuracy in the detection of this type of pest in tea leaves on a simulator.

The test at this stage is a result of predictions from data testing. The predictions obtained from the results of the classification by the method of convolutional neural network (CNN). Simulator testing first to fourth testing can be seen in Figure 12, 13, 14 and 15.

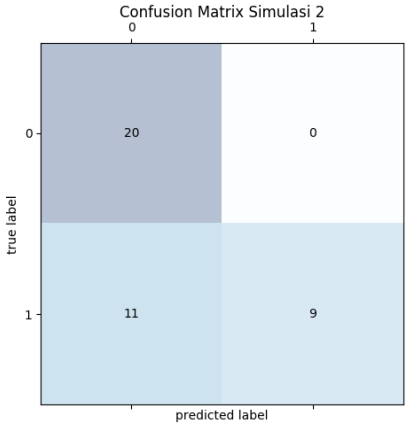


Figure 12. The First Testing

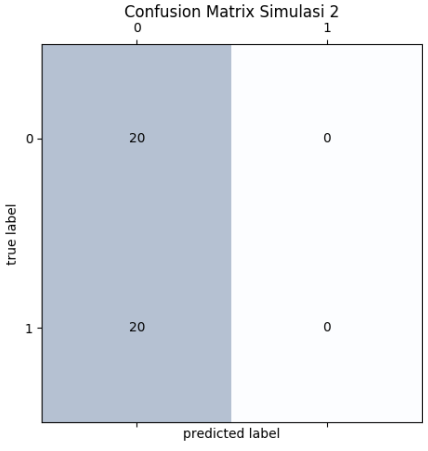


Figure 13. The Second Testing

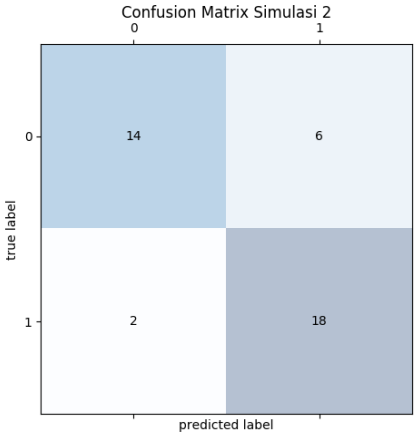


Figure 14. The Third Testing

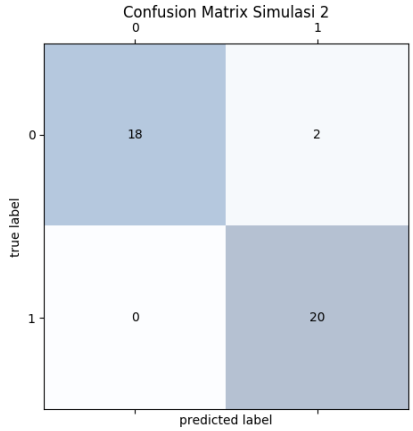


Figure 15. A Fourth Testing

2.8.4 Conclusion and Performance Testing Simulator

The conclusion of the testing simulations conducted indicate that simulation created to generate predictions of the kind of pest in tea leaves, from preprocessing, classification with convolutional neural network method (CNN) and the prediction of the results classification of CNN, already meet the functional requirements, but in the process still allows found shortcomings.

To the stage of the implementation of the processes and implementation of convolutional neural network method in detecting the type of pest in tea leaves. Prove that the processes and methods applied can run and implemented, by generating a predictive kind of pests with accuracy above 95% use epoch = 20. The process of testing the performance of the first until the last can be seen in table 11.

Table 11. Performance Test Results

Pengujian	*Epoch*	*Akurasi*
1	**5**	**75%**
2	**10**	**50%**
3	**15**	**80%**
4	**20**	**95%**

3 CLOSING

Conclusions and suggestions from research conducted in the implementation methods of convolutional neural network (CNN) in detecting the type of pest in tea leaves.

3.1 Conclusion

The results of the research that has been done, can be known pest type predictions based on the data of the image of hellopeltis pests and vermin blister blight. The results of the predictions in the form of the introduction of the type hellopeltis blister blight and pests.

Based on the results of implementation and testing, prediction accuracy the type of pest generated simulator is quite good with average accuracy value of 95% with epoch = 20. From the results obtained, it can be concluded that the use of methods of convolutional neural networks (CNN) can be implemented in the introduction image in detecting the type of pest in tea leaves based on image of pests and hellopeltis data blister blight the image, thus generating a prediction accuracy the type of pest that is good enough.

3.2 Advice

Based on the results of the research that has been done, there is still some further studies are needed in this research. As for suggestions for further research are:

1. In improving the accuracy of prediction, it needs to be studied more about the addition of aspects or other factors that affect the accuracy of the detection of the type of pest in tea leaves, such as the learning data factors, factors for introduction of data, and so on.
2. Needs to be studied more about the methods that can be applied to improve the accuracy of the detection of the type of pest in tea leaves.

REFERENCE

[1] Agustin, Soffiana and e. Joon, "Mango tree species Classification Gadung and Curut based on the texture of the leaves", ITS information system, 2011 Thesis.

[2] k. Muthukaman, p. Latha, r. p. and p. Selvi Nisha, "Classification Of Diseased Plant Leaves Using Neural Network ALgorithms", ARPN Journal of Engineering and Applied Scinces, vol. 10, no. 4, 2015.

[3] a. Maheswari. Bharathi, n.., p. Neelamegan., and t. Gayathridevi, "Classification and Recognition of the herb Leaf Using SVM Algorithm", Research Journal of Pharmaceutical, Biological and Chemical Sciences, pp. 0975-8585, 2014.

[4] p. Lestari, h. Bambang., and e. Susatio, "detection of Defects the tea leaves of Camellia Sinensis with Digital image processing and JST Learning Vector Quantization". Final Assignment. Telkom's University, 2011.

[5] a. Camuri and g. Volpe, "Gesture-Based Communication in Human-Computer Interaction". Pp. 978-3-540-21072-6, 2004.
[6] d. Hubel and Wiesel, t. "Receptive fields an functional architecture of monkey striate cortex cortex". Journal of Physiology. 1986 thesis.
[7] p. Sadowski, "Notes on Backpropagation", https://www.ics.uci.edu/~pjsadows/notes.pdf, accessed 6 July 2018.
[8] i. Wayan Suartika, a. Y Wijaya and r. Soelaiman, "Image Classification Using Convolutional Neural Network (Cnn) on the Caltech 101", ITS Engineering Journal, vol. 5, no. 1, 2016.
[9] r. Sebastian, "Python Machine Learnig", pp. 978-1-78355-513-0, September 2015.