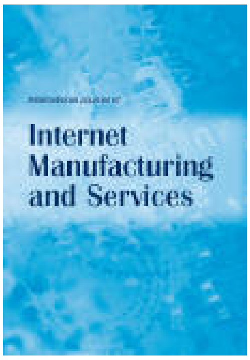


International Journal of Internet Manufacturing and Services

ISSN online: 1751-6056 - ISSN print: 1751-6048
https://www.inderscience.com/ijims

Utilisation of convolutional neural network on deep learning in predicting digital image to tree damage type

Rahmat Safe'i, Rico Andrian, Tri Maryono, Zuhri Nopriyanto

DOI: 10.1504/IJIMS.2024.10059666

Article History:

Received:	20 September 2022
Last revised:	26 June 2023
Accepted:	07 August 2023
Published online:	19 February 2024

Utilisation of convolutional neural network on deep learning in predicting digital image to tree damage type

Rahmat Safe'i*

Forestry Master Study Program,
Faculty of Agriculture,
University of Lampung,
Lampung, Indonesia
Email: rahmat.safei@fp.unila.ac.id
*Corresponding author

Rico Andrian

Department of Computer Science,
Faculty of Mathematics and Natural Science,
University of Lampung,
Lampung, Indonesia
Email: rico.andrian@fmipa.unila.ac.id

Tri Maryono

Department of Agrotechnology,
Faculty of Agriculture,
University of Lampung,
Lampung, Indonesia
Email: tri.maryono@fp.unila.ac.id

Zuhri Nopriyanto

Department of Computer Science,
Faculty of Mathematics and Natural Science,
University of Lampung,
Lampung, Indonesia
Email: zuhri.vhal@gmail.com

Abstract: Damage is often defined as a condition where there is a change in an object or shape. Damage does not only occur to objects, it can occur to living things, including trees. Damage to trees is seen in the physical shape of the tree that has changed shape, to find out it requires deep learning. One way that can be used is by modelling through computers through artificial intelligence, namely creating a deep learning model that can retrieve image information to recognise objects. This research aims to utilise convolutional neural network algorithm in deep learning to identify tree damage. The research stages carried out are input, feature extraction and classification, and output. The final result obtained is the successful identification of trees in deep learning on the model with an accuracy of 99.06% with a detection error of 0.94%. Detection errors occur due to similarities in terms of patterns, etc. This can be minimised by combining hyperparameters.

Keywords: computer vision; deep learning; forest health monitoring; FHM; type of tree damage; convolutional neural network; mobile-net.

Reference to this paper should be made as follows: Safe'i, R., Andrian, R., Maryono, T. and Nopriyanto, Z. (2024) 'Utilisation of convolutional neural network on deep learning in predicting digital image to tree damage type', *Int. J. Internet Manufacturing and Services*, Vol. 10, No. 1, pp.77–90.

Biographical notes: Rahmat Safe'i completed his Bachelor's degree in Forest Management, Faculty of Forestry, IPB (1999), Master's degree in the Forestry Science Study Program at the IPB Postgraduate Program (2005), and Doctoral degree in Forest Management Science, Pacsarjana School (2015). He has been active as a Lecturer in the Forestry Department, Faculty of Agriculture, University of Lampung, since 2006 and has been Head of the Master of Forestry Study Program from 2021 until now.

Rico Andrian completed his Bachelor's degree in the Computer Science Study Program, Faculty of Mathematics and Natural Sciences, Padjadjaran University, in 1999. He continued his master's education in the Computer Science Study Program, Faculty of Mathematics and Natural Sciences, Bogor Agricultural Institute, and graduated in 2013. Since 2005, he has been actively teaching as a Lecturer in the Computer Science Study Program at Lampung University.

Tri Maryono completed his Bachelor's degree in the Pest and Plant Disease Study Program, Faculty of Agriculture, University of Lampung (1999), his Master's degree at the Bogor Agricultural Institute (IPB) Postgraduate School in the Phytopathology Study Program (2011), and his Doctoral degree in the Agricultural Science Study Program, Faculty Gadjah Mada University (UGM) Agriculture with a Phytopathology specialisation (2019). He has been active as a Lecturer in the Department of Plant Protection, Faculty of Agriculture, University of Lampung, since 2005. Currently, he serves as secretary of the Department of Plant Protection, Faculty of Agriculture, University of Lampung.

Zuhri Nopriyanto is a student who has completed his undergraduate education in the Computer Science Study Program, Faculty of Mathematics and Natural Sciences, University of Lampung, in 2022.

1 Introduction

Forest health monitoring (FHM) is a way of monitoring forest health with human resources using a census method. The health of trees, which are forest ecosystems, is also monitored in this way, one of which is the type of damage to trees. The types of tree damage are a form of how trees are categorised as sick or healthy (Safe'i et al., 2022; Abimanyu and Safei, 2019). The tree parts (damage location) included in the observation element are the shoot's upper part and the root's bottom (Safe'i et al., 2019). To recognise damage requires observation by the eye, namely as a human visual, to identify whether there is damage or not on a tree, but this method has been done and requires many resources.

The types of damage identified to date total 16 types, according to previous research conducted using the FHM method (Safe'i et al., 2019). Determination of the type of damage based on the characteristics of the damage found on the tree. In this case, microorganisms, pathogens, and humans are also involved. Many agricultural industries are applying image processing and vision technologies to reduce equipment costs and increase computerisation (Mahajan et al., 2015). In facilitating these efforts, computerised science has one way: to create a deep learning model that can automatically retrieve information from digital images to recognise objects.

In this era, computerisation has played an enormous role in making solving problems more accessible, efficient, and effective. Hardware (e.g., cameras, lights, image archiving, and communication equipment) is the foundation of computer vision technology, while software (e.g., image processing algorithms) is the core of the system (Yin et al., 2022). However, traditional systems had poor processing effects and were lengthy and time-consuming (Yang et al., 2018). One of the technologies that can facilitate humans themselves is deep learning (Chung et al., 2020). With the development of artificial intelligence, continuous breakthroughs in deep learning have achieved great success in speech recognition, NLP processing, computer vision, video analysis, multimedia, etc. (Yu et al., 2019; Janabi et al., 2022). One advantage of deep learning is its ability to model nonlinear relationships (Zhao et al., 2020). This technology is very effective for processing raw data and creating patterns for decision-making purposes.

One of the methods in deep learning is convolutional neural network (CNN). In recent years, CNN has developed very rapidly in the field of vision recognition, completed many tasks at once, and become the backbone of research (Zhang et al., 2016). This deep-learning model is able to capture image data structures automatically, layer by layer, based on large amounts of data very well (Arpit et al., 2017). Deep learning recognises images based on input to the artificial neural network and then proceeds with an in-depth backpropagation algorithm to minimise loss of function (Tian, 2020). Furthermore, CNN can extract the connection and spatial information between its layers and express the relevant characteristics inside the image (Zhu et al., 2020). In CNN, there is a convolutional layer that functions as image learning to be more efficient in implementation. Therefore, researchers utilise this method of deep learning to facilitate human work in identifying the type of tree damage.

2 Review of literature

2.1 Leaf snap: a computer vision system for automatic plants species identification. Berlin: Springer-Verlag

Research conducted in 2012 by Neeraj Kumar and colleagues has proven that plant species can be identified automatically using computer vision (Kumar et al., 2012). Tests were conducted using an iPhone and Android OS system, which used the Nearest Neighbour method, and 184 plant samples from the Northeastern USA. The results showed that the plant species can be accurately identified up to 85.1% in 5.4 seconds for one plant species, and the system operates well despite using large datasets and labelled images.

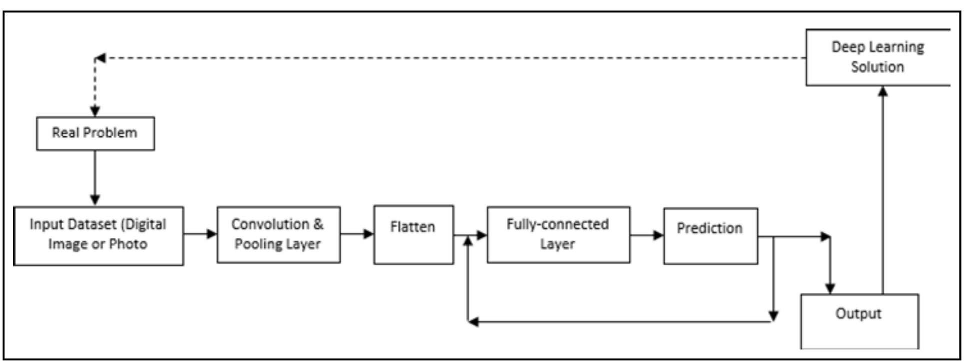
2.2 Road damage detection acquisition system based on deep neural network for physical asset management

This study was conducted by researchers from the University of Guadalajara, Mexico, in 2019. The research was related to the development of a deep neural network-based asphalt damage detection system. Asphalt damage samples were taken from several Italian and Mexican roads; the data used was in the form of videos and photos. The researchers trained objects to identify and classify road damage from images and videos in real-time with high accuracy and low intervention time using the digital asset management tool. This research was conducted with the aim of helping government agencies with infrastructure planning and maintenance (Angulo et al., 2019).

2.3 Application of forest health monitoring method in assessing tree damage in metro urban forests

This research took place from November to December 2018, and the main focus was to conduct research and analysis of tree damage in the urban forest of Metro City Stadium, Lampung, Indonesia. The purpose of the research was to determine the status of tree damage in the Metro City Stadium urban forest in the context of monitoring green open areas. The research was conducted using the FHM census method. The results obtained are the condition of damage: in the healthy category, there are 300 trees (77%), in the moderate category, there are 69 trees (18%), and in the sick category, there are 19 trees (5%) (Abimanyu and Safei, 2019).

Figure 1 Stage of research



3 Research method

The following are the stages of research conducted in predicting the type of tree damage using digital images using deep learning, shown in Figure 1.

Figure 2 Example of dataset (see online version for colours)

3.1 Input dataset (digital image or photo)

It is the initial stage of making the machine recognise what is in the image. For example, trees indicated to have damage are photographed from the front side of the damage. In computer technology systems, the quality of the input image will determine the results (output) of computer vision development in deep learning (Babatunde et al., 2015). For CNN to work well, the quality of the image as input will sustain the performance of the architecture (Acharya et al., 2017). With the influence of large-scale integrated circuits and programming, image processing system design was constantly innovating and improving (Tong et al., 2020). CNN: simultaneously learning the feature extraction layer and classification layer causes the output to be highly organised and highly dependent on the extracted input features (Laith et al., 2021). The image that the computer can see is an image's Red, Green, Blue (RGB) number. The value is between 0 and 255 for each colour in each image pixel.

The dataset input used in this study amounted to 1,600, with 100 photos or images in each class, where there are 16 classes. Images are taken by a camera with a certain resolution and will be processed by the pre-processing stage, namely the scaling process with a resolution of 224 × 224 pixels. This is necessary so that the computational process runs smoothly without eliminating the special characteristics of the image used as input data for further research. The division of datasets as input will be divided into three parts, namely training data, validation data, and testing data, with a ratio of 70:10:20.

Table 1 Dataset ratio

No.	*Dataset type*	*Ratio*	*Lots of data*
1	Training	70%	1,120
2	Validation	10%	160
3	Testing	20%	320
	Total of data:		*1,600*

3.2 Feature extraction and classification

Image data that indicates damage to the tree then enters the next stage, namely Feature Extraction. Generally, the types of layers in a CNN will be divided into two groups. CNN architecture with more layers has a central point to achieve a high level of accuracy but has the risk of being very complicated. The first layer is the feature extraction layer, located at the beginning of the architecture and composed of several layers; each layer is composed of neurons connected to the local region and the previous layer. The first type of layer is a convolutional layer, and the second layer is a pooling layer. Each layer makes adjustments to the existing parameters and filters to encourage increased performance (Reinel et al., 2021). The layers of the feature extraction layer are explained as follows:

3.2.1 Convolution layer

Image data that indicates damage to the tree then enters the next stage. Feature Convolution is the initial layer in the CNN algorithm in deep learning that will be passed by the input tree image data. This operation applies the output function to the feature map of the input image. This layer will extract, retrieve, and store the characteristics of the objects in the tree image.

3.2.2 Pooling layer

In this process, a filter with a specific size and stride is shifted across the feature map area. Several pooling methods include tree pooling, gated pooling, average pooling, min pooling, max pooling, global average pooling, and global max pooling (Laith et al., 2021). Pooling commonly used are max pooling and average pooling. The purpose of using a pooling layer is to reduce the dimension of the feature map (down-sampling), thus speeding up computation because there are fewer parameters to update and overfit (Dutta and Ghosh, 2021).

3.2.3 Flatten

At this stage, the value of the feature map array is converted into a vector. A vector form is a form that can be accepted as input at the fully connected layer stage. The change to vector form is intended so that when the tree damage image data enters the fully connected layer stage, each neuron or vector that has been formed is usually interconnected and can match one tree damage type feature with another tree damage type feature to be able to classify objects in tree damage images.

Table 2 16 types of damage in forest health monitoring (16 output layers)

No.	*Code*	*Data type*
1	01	Cancer
2	02	Conk
3	03	Open wound
4	04	Restenosis or Gnosis
5	05	Broken stems
6	06	Termite nest
7	11	Stems or roots broke with 3 feet
8	12	Stems or roots got broom
9	13	Broken or death roots
10	20	Liana
11	21	Loss or death of tree tops
12	22	Broken branches
13	23	Excessive broom
14	24	Leaves, shoots, and bud damaged
15	25	Leaf discoloration
16	31	And others
Number of classes: 16		

3.2.4 Fully-connected layer

This stage receives input in the form of one-dimensional data. The data in question contains the characteristics of the feature map that has changed its shape but does not change the contents of the characteristics stored in the feature map. The fully connected layer is a layer that acts as a classifier on the entire network on the CNN and is located in the last layer that connects (Zhang et al., 2019). A fully connected layer works by connecting each neuron formed by previous layers. Neurons are the characteristics of the type of tree damage (object) detected.

Neurons are connected so that the characteristics of the type of tree damage (object) stored can be set after CNN learns the characteristics in the previous layer. The characteristics of the type of tree damage (object) are set automatically because the neurons are interconnected. This stage also generated weights for each type of tree damage object that was successfully detected to be used as a reference and training material in other experiments.

3.2.5 Prediction

Prediction is the result after performing the fully connected layer process. Following the results of the previous analysis, these results will be forwarded to the CNN classification stage, where image prediction occurs.

The classes available in the labelling above are tree cancer, liana, bullet rust, and others, covering all types of tree damage in the table. Furthermore, a final example at that stage is an image of a tree, indicated by the type of tree cancer disease. The tree cancer entered the form of a class label that has the characteristics of the type of tree cancer damage itself. After going through several processes to the classification stage, the system will match the characteristics and predict the inputted image according to the respective class label, namely the tree cancer class label.

3.3 Output

Output is the final result after prediction, where the output result is a digital image identification consisting of 16 output layers, where 16 are the types of damage in FHM. The input image will output a classification result based on the 16 types of damage.

4 Experimental results and analysis

In this paper, we use tools in the form of Jupyter Notebooks, which run on a computer with Nvidia Tesla K20 specifications belonging to the software engineering laboratory at the University of Lampung. The implementation of deep learning with this algorithm uses MobileNet architecture version 1.00 written in Python with 1,600 digital images.

This research implements the TensorFlow and Keras libraries in the Python programming language because these libraries are developed explicitly for deep learning and machine learning (Yan et al., 2018). In the experiment, several customised input hyperparameters were used. The experimental site was an indoor laboratory in a university building, and the size was 9 m × 7 m.

There were four epochs used in the experiment. This study uses four epochs: 10, 30, 50, and 80. The four produced significant and different results at each epoch. The following describes the hyperparameter and summary model researchers used in this study, as shown in the Figure and table.

Table 3 Hyperparameter model

No.	*Parameter name*	*Value*
1	Batch sizes	32
2	Epoch	10, 30, 50, and 80
3	Pixel sizes	224 × 224 × 3
4	Learning rate	0.0001
5	Number of classes	16
6	Optimiser	Adam

Figure 3 Model summary

Model: "mobilenet_1.00_224"

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv1 relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormalization)	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2048
conv_pw_1_bn (BatchNormalization)	(None, 112, 112, 64)	256
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576
conv_dw_2_bn (BatchNormalization)	(None, 56, 56, 64)	256
conv_dw_2_relu (ReLU)	(None, 56, 56, 64)	0
conv pw 2 (Conv2D)	(None, 56, 56, 128)	8192
conv_pw_2_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_pw_2_relu (ReLU)	(None, 56, 56, 128)	0
conv_dw_3 (DepthwiseConv2D)	(None, 56, 56, 128)	1152
conv_dw_3_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_dw_3_relu (ReLU)	(None, 56, 56, 128)	0
conv_pw_3 (Conv2D)	(None, 56, 56, 128)	16384
conv_pw_3_bn (BatchNormalization)	(None, 56, 56, 128)	512
conv_pw_3_relu (ReLU)	(None, 56, 56, 128)	0
conv_pad_4 (ZeroPadding2D)	(None, 57, 57, 128)	0
conv_dw_4 (DepthwiseConv2D)	(None, 28, 28, 128)	1152
conv_dw_4_bn (BatchNormalization)	(None, 28, 28, 128)	512
conv dw 4 relu (ReLU)	(None, 28, 28, 128)	0
conv_pw_4 (Conv2D)	(None, 28, 28, 256)	32768
conv_pw_4_bn (BatchNormalization)	(None, 28, 28, 256)	1024
conv_pw_4_relu (ReLU)	(None, 28, 28, 256)	0
conv_dw_5 (DepthwiseConv2D)	(None, 28, 28, 256)	2304
conv_dw_5_bn (BatchNormalization)	(None, 28, 28, 256)	1024
conv_dw_5_relu (ReLU)	(None, 28, 28, 256)	0
conv_pw_5 (Conv2D)	(None, 28, 28, 256)	65536
conv_pw_5_bn (BatchNormalization)	(None, 28, 28, 256)	1024
conv_pw_5_relu (ReLU)	(None, 28, 28, 256)	0
conv_pad_6 (ZeroPadding2D)	(None, 29, 29, 256)	0
conv dw 6 (DepthwiseConv2D)	(None, 14, 14, 256)	2304
conv dw 6 bn (BatchNormalization)	(None, 14, 14, 256)	1024
conv_dw_6_relu (ReLU)	(None, 14, 14, 256)	0
conv_pw_6 (Conv2D)	(None, 14, 14, 512)	131072
conv_pw_6_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_pw_6_relu (ReLU)	(None, 14, 14, 512)	0

Model: "mobilenet_1.00_224"

Layer (type)	Output Shape	Param #
conv_dw_7 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_7_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_dw_7_relu (ReLU)	(None, 14, 14, 512)	0
conv pw 7 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_7_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_pw_7_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_8 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_8_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_dw_8_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_8 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_8_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_pw_8_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_9 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_9_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv dw 9 relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_9 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_9_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_pw_9_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_10 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_10_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_dw_10_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_10 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_10_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_pw_10_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_11 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_11_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv dw 11 relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_11 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_11_bn (BatchNormalization)	(None, 14, 14, 512)	2048
conv_pw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pad_12 (ZeroPadding2D)	(None, 15, 15, 512)	0
conv_dw_12 (DepthwiseConv2D)	(None, 7, 7, 512)	4608
conv_dw_12_bn (BatchNormalization)	(None, 7, 7, 512)	2048
conv_dw_12_relu (ReLU)	(None, 7, 7, 512)	0
conv_pw_12 (Conv2D)	(None, 7, 7, 1024)	524288
conv_pw_12_bn (BatchNormalization)	(None, 7, 7, 1024)	4096
conv_pw_12_relu (ReLU)	(None, 7, 7, 1024)	0
conv dw 13 (DepthwiseConv2D)	(None, 7, 7, 1024)	9216
conv dw 13 bn (BatchNormalization)	(None, 7, 7, 1024)	4096
conv_dw_13_relu (ReLU)	(None, 7, 7, 1024)	0
conv_pw_13 (Conv2D)	(None, 7, 7, 1024)	1048576
conv_pw_13_bn (BatchNormalization)	(None, 7, 7, 1024)	4096
conv_pw_13_relu (ReLU)	(None, 7, 7, 1024)	0

Total params: 3,228,864
Trainable params: 3,206,976
Non-trainable params: 21,888

As can be seen from Table 2, there are six hyperparameter values used in the research. The input image measuring 224 × 224 × 3 pixels will be processed using the CNN algorithm with Mobile-Net architecture version 1.00. A summary description of the model can be seen in Figure 2. The output of the image processing is the identification result in the form of 16 types of damage, according to FHM.

Table 4 Results model with mobile-Net v 1.00

No.	*Epoch*	*Training/validation*	*Time*
1	10	0.8821/0.8094	133s
2	30	1.0000/0.9781	392s
3	50	1.0000/0.9906	654s
4	80	1.0000/0.9844	1042s

Table 5 Confusion matrix with data testing

		X															
Y		*01*	*02*	*03*	*04*	*05*	*06*	*11*	*12*	*20*	*21*	*22*	*23*	*24*	*25*	*31*	*13*
	01	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	02	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	03	0	0	24	0	0	0	0	0	0	0	0	0	0	0	0	0
	04	0	0	0	26	0	0	0	0	0	0	0	0	0	0	0	0
	05	0	0	0	0	18	0	0	0	0	0	0	0	0	0	0	0
	06	0	0	0	0	0	22	0	0	0	0	0	0	0	0	0	0
	11	0	0	0	0	0	1	22	0	0	0	0	0	0	0	0	0
	12	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0
	20	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0
	21	0	0	0	0	0	0	0	0	0	18	0	1	0	0	0	0
	22	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0

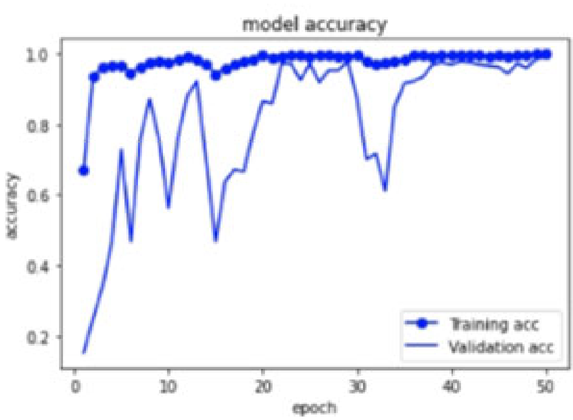
Notes: Description: X: As presumptive class (Prediction) Y: As ground truth class
01: Cancer
02: Conk
03: Open wound
04: Restenosis or Gnosis
05: broken stems
06: Termite nest
11: Stems or roots broke with 3 feet
12: root or stems brum
13: Broken or dead roots
20: Liana
21: Loss or death of tree tops
22: broken branches
23: Excessive Brum
24: Leaves, shoots, buds damaged
25: leaf discoloration
31: And others.

Table 5 Confusion matrix with data testing (continued)

								X								
Y	*01*	*02*	*03*	*04*	*05*	*06*	*11*	*12*	*20*	*21*	*22*	*23*	*24*	*25*	*31*	*13*
23	0	0	0	0	0	0	0	0	0	0	0	22	1	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	22	0	0
31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	17	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21

Notes: Description: X: As presumptive class (Prediction) Y: As ground truth class
01: Cancer
02: Conk
03: Open wound
04: Restenosis or Gnosis
05: broken stems
06: Termite nest
11: Stems or roots broke with 3 feet
12: root or stems brum
13: Broken or dead roots
20: Liana
21: Loss or death of tree tops
22: broken branches
23: Excessive Brum
24: Leaves, shoots, buds damaged
25: leaf discoloration
31: And others.

Figure 4 Model graphics on program (see online version for colours)



Finally, after the data is processed with the CNN algorithm, it produces four categories divided by the number of epochs used. There are two levels of accuracy: training and validation. Training accuracy is the accuracy used during the learning process, while validation is proof of the performance of the training process. The best validation accuracy is shown by the model with epoch 50, with a value of 0.9906 (99.06%) and a detection error rate of 0.0094 (00.94%). Detection errors are bound to occur due to misdetection both in features and errors in datasets that are similar in terms of patterns, colours, and others. This would be minimised by combining the hyperparameters and conducting repeated experiments.

To prove the effectiveness of the percentage of validation data accuracy with the mobile-net architecture above, a test was conducted with 320 digital images of available tree damage types. In this case, the model correctly identified 317 images of tree damage types, which can be calculated from the number of yellow cells, and three errors were shown in the red cells. The detection errors that occur are code 11, which is predicted to be code 6, code 21, which is predicted to be code 23, and code 23, which is predicted to be code 24. The use of codes is enforced to provide efficiency on the worksheet and can be seen again in Table 5.

5 Conclusions

FHM-based damage type prediction with digital images has been successfully carried out with the help of deep learning, CNN algorithm, and Mobile-Net architecture version 1.00. Satisfactory results were shown at epoch 50 with a detection error of 0.94% and a prediction success rate of 99.06%. It is expected that these results will have a positive impact on the world of computerisation. The limitations of this research are limited to the category of trees with or equal to 20 cm, the use of deep learning, namely only the CNN algorithm without other algorithms, and only including as many as 16 types of tree damage. The purpose of this research is to prove that the deep learning CNN algorithm with a certain architecture can identify the types of tree damage that have so far existed in as many as 16 types. The advantages of this research are that it can help obtain information related to tree damage that has not been known before in a computerised manner, assist analysts in examining trees precisely and accurately, and also provide existing references related to CNN and tree damage identification.

Acknowledgements

Thank you for funding the National Competitive Applied Research Scheme from the Directorate of Research, Technology and Community Service, Directorate General of Higher Education, Research, and Technology of the Ministry of Education, Culture, Research, and Technology by the Research contact Number: 114/E5/PG.02.00PT/2022 May 10, 2022.

References

Abimanyu, B. and Safei, R. (2019) 'Application of forest health monitoring method in assessing tree damage in metro city forest', *Bandar Lampung, Journal Sylva Lestari*, Vol. 7, No. 3, pp.289–298.

Acharya, U.R., Oh, S.L., Hagiwara, Y., Tan, J.H., Adam, M., Gertych, A. and Tan, R.S. (2017) 'A deep convolutional neural network model to classify heartbeats', *Comput. Biol. Med.*, October, Vol. 89, No. 1, pp.389–396.

Angulo, A.A., Vega-Fernández, J.A. Aguilar-Lobo, L.M., Natraj, S. and Ochoa-Ruiz, G. (2019) *Road Damage Detection Acquisition System Based on Deep Neural Network for Physical Asset Management*, MICAI, Mexico.

Arpit, D. et al., (2017) 'A closer look at memorization in deep networks', in *Proc. 34th Int. Conf. Mach. Learn.*, Vol. 6, No. 1, pp.233–242.

Babatunde, H.O., Armstrong, L., Leng, J. and Diepeveen, D. (2015) 'A Survey of computer- based vision systems for automatic identification of plant species', *Journal of Agricultural Informatics*, pp.61–71.

Chung, J., Choi, W., Park, J. and Ghosh, S. (2020) 'Domain wall memory-based design of deep neural network convolutional layers', *Journal IEEE Access*, January, Vol. 8, No. 3, pp.19783–19798.

Dutta, S. and Ghosh, S. (2021) 'Forest fire detection using combined architecture of separable convolution and image processing', *International Conference on Artificial Intellligence and Data Analytics*.

Janabi, A.H., Kanakis, T. and Johnson, M. (2022) 'Convolutional neural network based algorithm for early warning proactive system security in software defined networks', *Journal IEEE Access*, February, Vol. 10, No. 1, pp.14301–14310.

Kumar, N.P.N. (2012) *Leafsnap: A Computer Vision System for Automatic PlantSpecies Identification*, Springer-Verlag, Berlin.

Laith, A., Zhang, J., Humaidi, A.J., Al-Dujaili, A., Duan, Y., Al-Shamma, O. et al. (2021) 'Review of deep learning: concepts, CNN architecture, challenges, applications, future directions', *Journal of Big Data*, Vol. 8, No. 53, pp.1–74.

Mahajan, S., Das, A. and Sardana, H.K. (2015) 'Image acquisition techniques for assessment of legume quality', *Trends Food Sci. Technol.*, Vol. 42, No. 2, pp.116–133.

Reinel, T., Brayan, A., Alejandro, B., Alejandro, M., Daniel, A., Alejandro, A., Buenaventura, B., Simon, O., Gustavo, I. and Raul, R. (2021) 'GBRAS-Net: a convolutional neural network architecture for spatial image steganalysis', *Journal IEEE Access,* January, Vol. 9, pp.14340–14350.

Safe'i, R., Puspita, E.N. and Hilmanto, R. (2022) 'Assessment of tree vitality as an indicator of monitoring the health condition of community forest in agroforestry patterns', *Folia Forestalia Polonica*, Vol. 64, No. 4, pp.206–213, https://doi.org/10.2478/ffp-2022-002.

Safe'i, R., Pertiwi, D., Kaskoyo, H. and Indriyanto, (2019) 'Identification of tree damage condition using forest health monitoring method in Tahura WAR, Lampung Province', *Perennial Journal*, Vol. 15, No. 1, pp.1–7 ISSN: 1412 7784.

Tian, Y. (2020) 'artificial intelligence image recognition method based on convolutional neural network algorithm', *Journal IEEE Access Digital Object Identifier*, July, Vol. 8, pp.125731–125744.

Tong, X-Y., Guo, C. and Cheng, H. (2020) 'Multi-source remote sensing image big data classification system design in cloud computing environment', *Int. J. Internet Manufacturing and Services*, Vol. 7, Nos. 1–2, pp.130–145.

Yan, Y., Zhang, L., Li, J., Wei, W. and Zhang, Y. (2018) 'Accurate spectral super-resolution from single RGB image using multi-scale CNN', in *Proc. Springer Chin. Conf. Pattern Recognit. Comput. Vis. (PRCV)*, pp.206–217.

Yang, J., Jiang, B. and Song, H. (2018) 'A distributed image-retrieval method in multi-camera system of smart city based on cloud computing', *Future Generation Computer Systems*, Vol. 81, No. 12, pp.244–251.

Yin, H., Yi, W. and Hu, D. (2022) 'Computer vision and machine learning applied in the mushroom industry: a critical review', *Journal Computers and Electronics in Agriculture*, July, Vol. 198, p.107015.

Yu, J., Zheng, X. and Wang, S. (2019) 'A deep autoencoder feature learning method for process pattern recognition', *J. Process Control*, Vol. 79, pp.1–15.

Zhang, C., Bengio, S., Hardt, M., Recht, B. and Vinyals, O. (2016) 'Understanding deep learning requires rethinking generalization', arXiv:1611.03530 [online] http://arxiv.org/abs/1611.03530 (accessed 1 September 2022).

Zhang, Y-D., Dong, Z., Chen, X., Jia, W., Du, S., Muhammad, K. and Wang, S-H. (2019) 'Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation', *Journal Multimedia Tools Appl*., February, Vol. 78, No. 3, pp.3613–3632.

Zhao, L.L., Wang, B., Mbachu, J. and Egbelakin, T. (2020) 'Using artificial neural networks to forecast producer price index for New Zealand', *Int. J. Internet Manufacturing and Services*, Vol. 7, No. 3, pp.191–215.

Zhu, F., Zhang, D., Wang, S., Wang, Y., Cheng, X., Huang, Z. and Liu, Y. (2020) 'Understanding place characteristics in geographic contexts through graph convolutional neural network', *Ann. Amer. Assoc. Geographers*, Vol. 110, No. 2, pp.408–420.