IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Mathematical Modelling of Engineering Problems
Vol. 11, No. 5, May, 2024, pp. 1279-1285
Journal homepage: http://iieta.org/journals/mmep

Identification of Rice Plant Diseases Using Convolutional Neural Network Method

Kusworo Adi[1*], Catur Edi Widodo[1], Aris Puji Widodo[2], Agus Setiadi[3], Aji Setiawan[4]

[1] Department of Physics, Faculty of Science and Mathematics, Diponegoro University, Semarang 50275, Indonesia
[2] Department of Informatics, Faculty of Science and Mathematics, Diponegoro University, Semarang 50275, Indonesia
[3] Department of Agriculture, Faculty of Animal and Agricultural Sciences, Diponegoro University, Semarang 50275, Indonesia
[4] Department of Information Technology, Faculty of Engineering, Darma Persada University, East Jakarta 13450, Indonesia

Corresponding Author Email: kusworoadi@lecturer.undip.ac.id

https://doi.org/10.18280/mmep.110517

Received: 26 December 2023
Revised: 27 February 2024
Accepted: 5 March 2024
Available online: 30 May 2024

Keywords:
rice disease, Convolutional Neural Network (CNN), Android, image classification, climate change impact

ABSTRACT

Rice plants (Oryzae Sativa sp.) are rice-producing food plants which play an important role in economic life in Asian countries, especially Indonesia. This primary need is very difficult to replace with other staples, such as corn, tubers, sago and other sources of carbohydrates. However, global climate change which has an impact on climate anomalies causes rice diseases to develop rapidly. The aim of this research is to create an application on an Android-based cell phone that can identify and classify rice plant diseases using the Convolutional Neural Network (CNN) method. The research material was 1600 images of rice leaves consisting of 4 classes, namely 400 images of rice affected by Leaf Blight disease, 400 images of rice affected by Brown Spot disease, 400 images of rice affected by Leaf Smut disease, and 400 images of healthy rice. In each class, the images are divided into 380 for training and 20 for testing. During the rice image training process using the Python programming language, the accuracy results were 83%. Then the results are saved in the form of a model file, and entered as training data into the program in Android Studio to be used as an application. Testing the application with 20 rice leaf images for each class resulted in an actual accuracy of 94%. This application can also be run on all versions of Android.

1. INTRODUCTION

Rice is a vital food crop in Southeast Asia, particularly in Indonesia, where it is utilized in daily life. It is particularly challenging to substitute rice for other essential foods in the context of a staple diet. Included among these are corn, tubers, sago, and various other types of carbohydrates. So, rice becomes the main staple of society in meeting primary needs and main carbohydrate intake, which is easily converted into energy [1]. Climate greatly influences the development of rice plants, but global climate change causes the development of pests and diseases that threaten the safety of rice production. As a result, farming profits decrease because they have to reduce the costs of controlling pests and diseases, which are increasingly high, and the quality of production decreases so that they cannot compete in the market [2, 3].

In Indonesia, diseases in rice plants are bacterial leaf blight (Xanthomonas campestris pv. Oryzae), tungro disease (tungro virus), pyricularia leaf spot (Pyricularia grisea), stem rot (Helminthosporium sigmodeun), leaf sheath blight (Rhizoctonia solani Kuhn), regen stunt and grassy stunt. However, detecting rice disease is very difficult because of the limited range of vision due to the small physical shape of rice [4, 5]. Therefore, technology that is able to detect rice diseases automatically is needed. The classical approach to detecting diseases in rice plants is conducted through various segmentation techniques, including the utilization of color segmentation such as Hue, Intensity, and Saturation (HIS). However, it still faces challenges due to the unstable classification values attributed to classical image processing, which necessitates consistent dimensions and lighting conditions [6].

Deep Learning is a technological development that has many capabilities and has become very popular in the scientific literature because of its capability to complete large amounts of data. The Convolutional Neural Network (CNN) approach is one such way. The Convolutional Neural Network technique is an advancement over the Multilayer Perceptron (MLP), an architecture originally developed for two-dimensional data processing. CNN is classified as a Deep Neural Network because of its extensive network depth and its extensive application in the processing of picture data [7-13]. However, apart from the advantages of CNN, there are also disadvantages, namely that it requires a Graphic Processing Unit (GPU) so that the process of creating a training model is faster without any problems [14-19].

In order to make farmers' work easier, tools are needed that have been inserted into the program to detect and identify in real-time whether rice is affected by disease or not. However, these tools can also be carried anywhere efficiently and are compatible, for example, smartphones. Based on this background, in this research, we created a disease

identification application in rice using the CNN method assisted by the Python programming language, which was implemented on an Android smartphone with the help of Android Studio software. The utilization of smartphones in certain cases has been demonstrated to facilitate workers in detecting objects that are the subject of research issues, such as in the detection of Cercospora Leaf Spot (CLS) disease on sugar beet leaves [20].

In order to facilitate comprehension for readers, we have outlined the paper's structure as follows. Section 1, namely the introduction, provides a glimpse of rice diseases and the need for automatic detection methods. Section 2 provides more detailed knowledge about rice diseases whose characteristics can be seen through the leaves and stems. In Section 3, the methodology, the research materials and particular classification procedures are delineated. These include the acquisition of data for training and testing purposes, the determination of the CNN architecture, and the execution of the classification process. In addition, verification of the method is covered in this part. The discussion and results sections present a comprehensive summary of the evaluation measures that were employed to appraise the suggested method's performance. In this section, the accuracy of the proposed approach will be evaluated.

Additionally, difficulties and constraints found throughout the testing process will be elaborated upon. This will be elaborated upon in Section 4 of the present article. Section 5 will provide the conclusions.

2. RICE DISEASES

In cultivating rice plants, we cannot be separated from the threat of diseases that often attack these plants. If disease control is not done properly, it can reduce the productivity of the rice plant. Therefore, farmer knowledge is needed to be able to recognize the types of rice plant diseases so that farmers can identify and apply control appropriately, quickly, and accurately [4, 5]. Some diseases that generally attack rice plants are brown spots, leaf blight, and leaf smut. Examples of rice diseases can be seen in Figure 1.

(a) (b) (c)

Figure 1. Rice disease (a) bacterial leaf blight, (b) leaf smut, (c) brown spot

Leaf blight is caused by the bacteria Xanthomonas campestris pv. Oryzae, which is spread in almost all rice-growing areas, both in the lowlands and highlands. Diseases appear in the dry and rainy seasons. Yield losses caused by bacterial leaf blight can reach 60%. The heaviest damage occurs when the disease attacks young plants, which can cause the plants to die. The main characteristic of this disease is the appearance of spot-shaped lesions on the leaves of rice plants. These lesions can be brown to black in color with edges that are usually lighter.

The fungus Ustilaginoidea virens causes leaf smut. This disease appears to be one of the diseases that is very detrimental to rice crops. The development of this disease is aided by rainy weather conditions and high relative humidity, low temperatures, and wind, which transmits spores from plant to plant. Rainfall followed by cloudy days during flowering may favour the disease. During a serious attack, the level of damage can reach 80%. Symptoms of the disease are clearly visible when the affected rice grains are filled with spores and have an abnormal shape. Rice grains become large lumps that are green on the outside, yellow on the inside, and 2-3 times the size of a rice grain.

The fungus causes brown spot Helminthosporium sp. Brown spot can cause the death of young plants and reduce grain quality. This fungus has a wide range of hosts and is easily spread. This disease causes production losses of up to 90%. The symptoms are that oval to round spots appear on the leaves. The spots are dark brown with light brown edges.

3. METHODS

3.1 Dataset and tolls

The dataset from this research is 4 classes of rice images with a total of 1600 images obtained from kaggle.com. This rice image dataset includes 400 images of rice affected by Leaf Blight disease, 400 images of rice affected by Brown Spot disease, 400 images of rice affected by Leaf Smut disease, and 400 images of healthy rice. Each class is divided into 380 (95%) training data and 20 (5%) test data. Detailed distribution of the dataset can be seen in Table 1.

Table 1. Explanation of the dataset

Category	Number of Images	Training	Testing
Leaf Blight	400	380	20
Brown Spot	400	380	20
Leaf Smut	400	380	20
Healthy	400	380	20

The tools used in this research are computers, Android-based cell phones, and the programming languages Python, Kotling, and Java. The computer used in making this program has the following specifications: AMD Ryzen 7 4800H chipset, 16 GB DDR4 RAM, 6 GB NVIDIA RTX 2060 GPU, 512 GB SSD Storage, and Windows 10 Students OS. The cell phone used to implement this system has the following specifications: Qualcomm SM8250 Snapdragon 865 chipset, Adreno 650 GPU, Android OS version 11, 256GB ROM storage, and 8 GB RAM. Python is employed for the purpose of teaching and testing, whereas Java is utilized for the development of applications for mobile devices.

3.2 CNN model used

CNN is a feed-forward neural network that is composed of many layers, generally 20-30 layers. Convolutional Neural Networks have a distinctive characteristic, namely the

existence of a convolutional layer, which is used to carry out the recognition process of the image to be processed. The discovery of CNN was first made by Hubel and Wiesel regarding the virtual cortex in cats sense of sight [21, 22]. CNN is advancing with the availability of several pre-trained learning models aimed at enhancing accuracy levels based on previously established models. Some of these models include VGG16, ResNet [23], MobileNet [24], and DenseNet [25].

CNN technically use a sequence of operations, namely convolutional layers, to train and evaluate each input image. Subsequently, features are extracted from successive input images using pooling. Following the pooling operation, the image is flattened prior to its input into the fully connected layer process, which is responsible for performing the classification task. The implementation of the CNN method is illustrated in Figure 2.

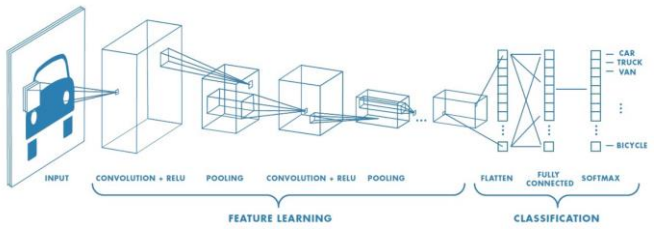

Figure 2. Process of performing CNN

The network architecture employed in this study is based on the AlexNet framework. To address the challenges encountered during the training process for high-resolution image data, it is imperative to establish multiple layers and investigate training data parameters. This will enable the CNN model to deliver satisfactory performance and mitigate gradient instability, which is particularly prevalent in deep networks [26, 27].

There are two crucial steps in the CNN process, namely, the training and testing phases. The training phase is where the CNN is trained to achieve high accuracy in classification. Meanwhile, testing is the classification process using the weights and biases obtained from the training process. This process is similar to training, except there is no backpropagation after the feedforward process. Thus, the outcome of this process yields the accuracy of the classification, data that was unsuccessfully classified, image numbers that failed to be classified, and the network structure formed from the feedforward process.

3.3 Training

The first step taken in this research was to divide the data into two, namely, data for training and data for testing. Then, continue to carry out the convolution process on the image of the rice disease. The image of rice will be divided into multiple low-resolution images. Subsequently, input each convolved image of rice sickness into a diminutive neural network. After the convolution is completed, each little image fragment is utilized as input to generate a feature representation. This functionality enables CNN to identify diseases on rice leaves based on an image of the location of the diseased leaf. Weight sharing occurs in a neural network, and this operation is executed uniformly across all components of an image by employing a single filter. The relevant portion of the image of the rice leaf disease will be annotated as an object of interest if an image of interest is detected.

3.4 Testing

After training images of disease in rice using the CNN method, the next step is to carry out testing by entering the reduced array results into the neural network in model form and seeing the level of accuracy achieved by the CNN model. The neural network will decide whether the image is suitable or not. This step is called a fully connected network. If the accuracy level is below 70%, then the error rate in classification is high, and retraining must be carried out. However, if the accuracy level is above or equal to 70%, then the training process is stopped, and testing is continued using testing data (5% of the total data). From this testing process, accuracy, precision, recall, and F-1 score can be measured. With the results of the model that has been trained and tested, the rice disease identification system can be used on Android cell phone devices.

4. RESULT AND DISCUSSION

In this chapter, the results of the research will be presented in accordance with the research procedures described previously, starting with collecting a dataset of images of healthy and sick rice, carrying out training with CNN, creating a model for Android, and ending with analyzing the research results. Examples of four classes of rice image datasets that will be trained can be seen in Figure 3.

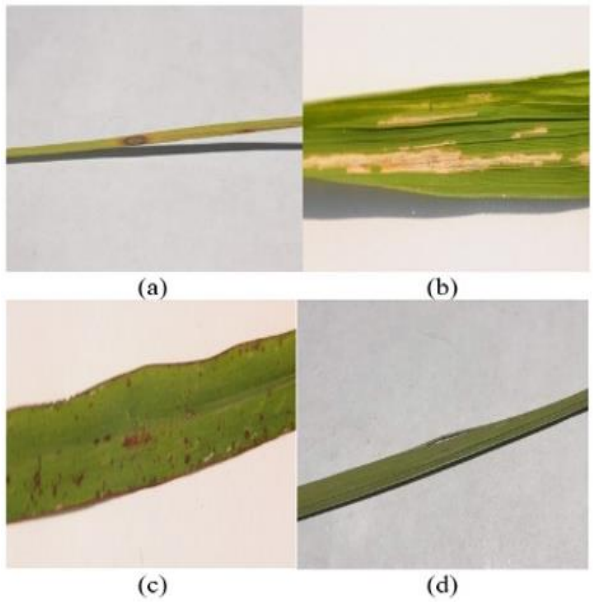

Figure 3. Examples of four classes of rice image datasets that will be trained, namely (a) brown spot, (b) bacterial leaf blight, (c) leaf smut, and (d) healthy rice leaves

This research began by preparing 4 classes of image datasets, including leaf spots (Figure 3(a)), brown spots (Figure 3(b)), hispa (Figure 3(c)), and healthy rice leaves (Figure 3(d)). The step continues by calling the command to set the directory that will be used as material for training. Then call opencv with the command word cv2 to resize the image to a length of 128 pixels, a height of 128 pixels and a thickness of 3.

After resizing, continue changing the image dataset into an array by calling numpy as np. Next, divide the image dataset into two, namely the image that will be trained and the image that will be tested. The next stage is to call Tensorflow to

import ImageDataGenerator and to_categorical to set the images and image categories to be trained. Then create parameters for the model in training by calling Tensorflow to import Sequential, Dense, Adam, Dropout, Flatten, Conv2D, and MaxPooling2D. This process is carried out by entering a total of 32 filters and a kernel size of 3×3 which is then inserted with Max Pooling to reduce the spatial dimensions of the output volume. The value entered in Max Pooling is 2. The convolution process continues, but the total filter value is reduced to 16, the second kernel size is 1×1, and the Max Pooling size is 2×2. The dense value is filled with 2048 neurons. Parameters can be seen in Figure 4.

```
def myModel():
    noOfFilters = 32
    sizeOfFilters1 = (3,3)
    sizeOfFilters2 = (1,1)
    sizeOfPool = (2,2)
    noOfNode = 2048
```

Figure 4. Filter parameters in creating a training model

The convolution process also includes a 'reLU' activation function. In the second Dense part of the process, the 'softmax' activation function is entered. Then, add the optimizers parameter Adam with learning_rate = 0.001, loss rate = 'categorical_crossentropy', and metrics = accuracy. Next, run it with a sample size (batch size) of 32 and set the epochs 20 times. The model run using the terminal can be seen in Figure 5. Using GPU, the training process only takes 5 minutes. The accuracy level results were 83%, as in Figure 6, while the failure (loss) results were 42%, as in Figure 7. The accuracy level results exceeded the specified target, namely above 75%.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 60)        1560
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 24, 24, 60)        90060
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 12, 12, 60)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 10, 10, 30)        16230
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 30)          8130
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 4, 4, 30)          0
_________________________________________________________________
dropout (Dropout)            (None, 4, 4, 30)          0
_________________________________________________________________
flatten (Flatten)            (None, 480)               0
_________________________________________________________________
dense (Dense)                (None, 500)               240500
_________________________________________________________________
dropout_1 (Dropout)          (None, 500)               0
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 2004
=================================================================
Total params: 358,484
Trainable params: 358,484
Non-trainable params: 0
```

Figure 5. Model to be trained using the CNN method

After carrying out the training process, a model of the training results is created, which will be inserted into the Android application with the help of Android Studio software. The programming language used to identify rice diseases uses kotling. The training model was created with the 'tflite' extension to make the import process easier for identification projects. In making an Android application, the most important thing is to activate the camera. Before activating, you must create access permissions to the camera so that it is not automatically denied.

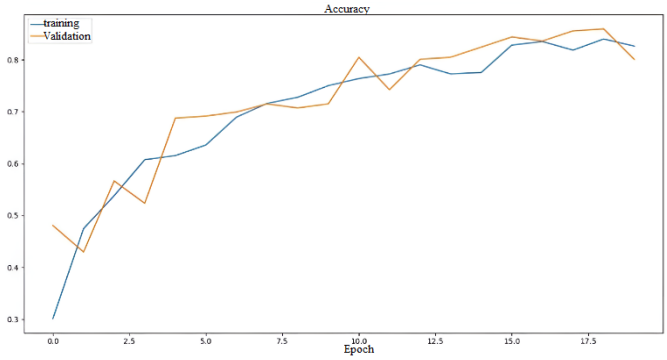

Figure 6. The graph shows that the level of accuracy at the peak point was found to be 88%

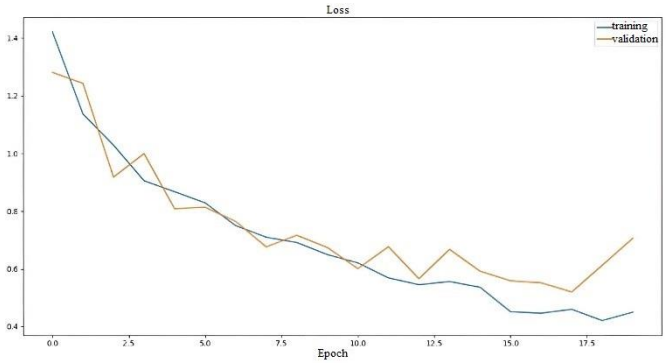

Figure 7. The graph shows a failure rate (loss) of 42% after the training process

After activating the camera, then enter the analysis program, which will be inserted into the camera. This analysis program functions to determine the output results from the camera. Then, enter the model that has been trained previously. Then, convert the bitmap image into a tensor image (TensorImage). Continue processing the image sorting based on the results of the training model, and the output will be placed at the top. Finally, convert the probability results into an identification list in the form of percentages. This percentage will reflect the image of rice affected by disease via the rear camera of a cell phone in real time.

After all the program codes have been input, the Android application for rice disease identification can be created. Making this application also pays attention to the API (Application Programming Interface) level so that it is easy to use on all versions of Android-based cell phones without any mistakes or errors, both during installation and use of the application. In testing this application, 20 test images were taken for each class from the dataset that was created. This test is carried out to determine actual performance metrics in the form of percentages such as accuracy, precision, recall, and F1 score. It can be written as follows:

a) Accuracy denotes the degree to which the model classifies instances accurately.

$$acc=\frac{TP+TN}{TP+FP+TN+FN} \quad (1)$$

b) Precision pertains to the degree of accuracy that

distinguishes the results generated by the model from the desired data.

$$prec = \frac{TP}{TP+FP} \quad (2)$$

c) The recall metric quantifies the effect of the model on information retrieval.

$$rec = \frac{TP}{TP+FN} \quad (3)$$

d) The F-1 score quantifies the relationship between recall and weighted average precision.

$$F\text{-}1\ score = \frac{2*rec*prec}{rec+prec} \quad (4)$$

where, True Positive (TP): We predict positive, and actually, it is actually positive. True Negative (TN): we predict negative, and the actual result is actually negative. False Positive (FP): We predict positive, and it turns out the prediction is wrong, but the actual result is negative. False Negative (FN): We predict negative, and it turns out the prediction is wrong, but the actual result is positive. In this research, because there are four classes, positive is defined as those that fit the category and negative is defined as those that do not fit the category. The test results can be seen in the confusion matrix in Table 2.

Table 2. Confusion matrix

Predictive Value	Actual Value			
	Leaf Blight	Leaf Smut	Brown Spot	Normal
Leaf Blight	19	0	0	2
Leaf Smut	0	20	2	0
Brown Spot	0	0	18	0
Normal	1	0	0	18

From the confusion matrix in Table 2, accuracy, precision, recall, and F-1 score can be calculated using formulas (1) to (4) with adjustments for four categories:

$$acc = \frac{19+20+18+18}{80} \times 100\% = 94\%$$

$$precision \text{ for leaf blight} = \frac{19}{21} \times 100\% = 90\%$$

$$precision \text{ for leaf smut} = \frac{20}{22} \times 100\% = 90\%$$

$$precision \text{ for brown spot} = \frac{18}{18} \times 100\% = 100\%$$

$$precision \text{ for normal} = \frac{18}{19} \times 100\% = 95\%$$

$$precision \text{ (average)} = \frac{90\%+90\%+100\%+95\%}{4} = 94\%$$

$$recall \text{ for leaf blight} = \frac{19}{20} \times 100\% = 95\%$$

$$recall \text{ for leaf smut} = \frac{20}{20} \times 100\% = 100\%$$

$$recall \text{ for brown spot} = \frac{18}{20} \times 100\% = 90\%$$

$$recall \text{ for normal} = \frac{18}{20} \times 100\% = 90\%$$

$$recall \text{ (average)} = \frac{95\%+100\%+90\%+90\%}{4} = 94\%$$

$$F\text{-}1\ score = \frac{2*(94\%*94\%)}{94\%+94\%} = 94\%$$

From the results of the performance measurements, it can be seen that the accuracy, precision, recall, and F-1 scores produce the same value, namely, 94%. This shows that it can predict all class categories with the same performance. What's interesting is that predictions for brown spots have 100% accuracy, meaning all brown spots are predicted correctly. On the other hand, predictions for leaf smut have a recall of 100%, meaning that everything that is predicted to be leaf smut is really leaf smut.

To install the system on an Android cell phone, a model was created using the TensorFlow Lite Maker add-on with the Jupyter Notebook software. This software can be obtained by the website: https://repo.anaconda.com/archive/Anaconda3-2021.05-Windows-x86_64.exe. Then, download it by opening Anaconda and typing pip install Jupyter Notebook and pip install Tensorflow Lite Maker.

After the model is created, it can be imported with Android Studio. However, a template for using an Android camera is needed so that rice disease detection can be used. The template can be downloaded via the command prompt by typing git clone https://github.com/hoitab/TFLClassify.git. Once downloaded, in Android Studio, press file -> open, which then looks for the 'TFL Classify' folder. If you have found the folder, press 'ok'. Android Studio will automatically create Gradle so that the model that has been created can be included in Android Studio data. Before the model can be implemented into an Android cell phone, make sure the phone is connected to the computer first. The model can be entered in the project window by right-clicking the start folder -> New -> Other -> Tensorflow Lite Model. After the training model has been entered, the Android application installation process can be carried out by pressing the start button. Android applications can also be created by pressing the build section -> build Bundle / APK -> build APK. If this system has been installed on an Android cell phone, it can be used to identify rice disease using the cell phone's camera system. Figure 8 shows an example of an interface screen when using the system where the system can identify brown spot disease. By using this system, farmers can easily identify rice diseases independently, without the need to use expert services, thereby saving time and consultation costs. Moreover, in the vast country of Indonesia, the availability of experts is very limited in rural areas far from cities.

Figure 8. Example of system interface

5. CONCLUSIONS

In this study, we investigate a convolution neural network-based approach for detecting rice diseases. We use a convolution neural network to detect images of rice leaves, whether they are healthy or have Leaf Blight, Brown Spot or Leaf Smut disease identified. Training was carried out using 380 images for each type of disease, and testing was carried out using 20 images for each type of disease. The results show that our proposed method works well with 94% accuracy, 94% precision, 94% recall and 94% F-1 score. Based on the results obtained in this study, we conclude that rice farmers can use the method we propose to detect rice diseases independently without the help of an expert. For future research, we hope to increase accuracy and reduce errors in training results. Furthermore, we will also develop this research with more types of rice diseases.

FUNDING

The paper was funded by the Diponegoro University, Semarang (Grant No.: 609-84/UN7.D2/PP/VIII/2023).

REFERENCES

[1] Saragih, B. (2001). Keynote address ministers of agriculture government of Indonesia. In 2nd National Workshop on Strengthening the Development and Use of Hibrid Rice in Indonesia, Indonesia, pp. 1-10.

[2] Pretty, J., Pervez Bharucha, Z. (2015). Integrated pest management for sustainable intensification of agriculture in Asia and Africa. Insects, 6(1): 152-182. https://doi.org/10.3390/insects6010152

[3] Dara, S.K. (2019). The new integrated pest management paradigm for the modern age. Journal of Integrated Pest Management, 10(1): 12. https://doi.org/10.1093/jipm/pmz010

[4] Singh, R. (2016). Brown spot of rice: An overview.

[5] Nagendran, K., Karthikeyan, G., Peeran, M.F., Raveendran, M., Prabakar, K., Raguchander, T. (2013). Management of bacterial leaf blight disease in rice with endophytic bacteria. World Applied Sciences Journal, 28(12): 2229-2241. https://doi.org/10.5829/idosi.wasj.2013.28.12.2009

[6] Ramesh, S., Vydeki, D. (2020). Recognition and classification of paddy leaf diseases using Optimized Deep Neural network with Jaya algorithm. Information Processing in Agriculture, 7(2): 249-260. https://doi.org/10.1016/j.inpa.2019.09.002

[7] Pantazi, X.E., Moshou, D., Tamouridou, A.A. (2019). Automated leaf disease detection in different crop species through image features analysis and One Class Classifiers. Computers and electronics in agriculture, 156: 96-104. https://doi.org/10.1016/j.compag.2018.11.005

[8] Phadikar, S., Sil, J., Das, A.K. (2013). Rice diseases classification using feature selection and rule generation techniques. Computers and Electronics in Agriculture, 90: 76-85. https://doi.org/10.1016/j.compag.2012.11.001

[9] Lu, Y., Yi, S., Zeng, N., Liu, Y., Zhang, Y. (2017). Identification of rice diseases using deep convolutional neural networks. Neurocomputing, 267: 378-384. https://doi.org/10.1016/j.neucom.2017.06.023

[10] Pothen, M.E., Pai, M.L. (2020). Detection of rice leaf diseases using image processing. In 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, pp. 424-430. https://doi.org/10.1109/ICCMC48092.2020.ICCMC-00080

[11] Ahmed, K., Shahidi, T.R., Alam, S.M.I., Momen, S. (2019). Rice leaf disease detection using machine learning techniques. In 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI), Dhaka, Bangladesh, pp. 1-5. https://doi.org/10.1109/STI47673.2019.9068096

[12] Azim, M.A., Islam, M.K., Rahman, M.M., Jahan, F. (2021). An effective feature extraction method for rice leaf disease classification. Telkomnika (Telecommunication Computing Electronics and Control), 19(2): 463-470. https://doi.org/10.12928/telkomnika.v19i2.16488

[13] Ferentinos, K.P. (2018). Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture, 145: 311-318. https://doi.org/10.1016/j.compag.2018.01.009

[14] Paithane, P.M., Kakarwal, S.N. (2022). Automatic pancreas segmentation using a novel modified semantic deep learning bottom-up approach. International Journal of Intelligent Systems and Applications in Engineering, 10(1): 98-104. https://doi.org/10.18201/ijisae.2022.272

[15] Liang, W.J., Zhang, H., Zhang, G.F., Cao, H.X. (2019). Rice blast disease recognition using a deep convolutional neural network. Scientific Reports, 9(1): 1-10. https://doi.org/10.1038/s41598-019-38966-0

[16] Sethy, P.K., Barpanda, N.K., Rath, A.K., Behera, S.K. (2020). Deep feature based rice leaf disease identification using support vector machine. Computers and Electronics in Agriculture, 175: 105527. https://doi.org/10.1016/j.compag.2020.105527

[17] Sony, A. (2019). Prediction of rice diseases using convolutional neural network (in Rstudio). International Journal of Innovative Science and Research Technology, 4(12): 595-602.

[18] Salini, R., Farzana, A.J., Yamini, B. (2021). Pesticide suggestion and crop disease classification using machine learning. International Journal of Engineering Science and Computing, 63(5): 9015-9023.

[19] Kakarwal, S., Paithane, P. (2022). Automatic pancreas segmentation using ResNet-18 deep learning approach. System Research and Information Technologies, 2: 104-116. https://doi.org/10.20535/SRIT.2308-8893.2022.2.08

[20] Adi, K., Widodo, C.E., Widodo, A.P., Margiati, U. (2022). Detection of foreign object debris (Fod) using convolutional neural network (CNN). Journal of Theoretical and Applied Information Technology, 100(1): 184-191.

[21] Fuad, M.S., Anam, C., Adi, K., Dougherty, G. (2021). Comparison of two convolutional neural network models for automated classification of brain cancer types. In AIP Conference Proceedings, 2346(1): 040008. https://doi.org/10.1063/5.0047750

[22] Matin, M.M.H., Khatun, A., Moazzam, M.G., Uddin, M.S. (2020). An efficient disease detection technique of rice leaf using AlexNet. Journal of Computer and Communications, 8(12): 49-57.

https://doi.org/10.4236/jcc.2020.812005
[23] Chen, J., Zhang, D., Nanehkaran, Y.A., Li, D. (2020). Detection of rice plant diseases based on deep transfer learning. Journal of the Science of Food and Agriculture, 100(7): 3246-3256. https://doi.org/10.1002/jsfa.10365
[24] Bi, C., Wang, J., Duan, Y., Fu, B., Kang, J. R., Shi, Y. (2022). MobileNet based apple leaf diseases identification. Mobile Networks and Applications, 27(1): 172-180. https://doi.org/10.1007/s11036-020-01640-1
[25] Liu, Z., Wu, J., Fu, L., Majeed, Y., Feng, Y., Li, R., Cui, Y. (2019). Improved kiwifruit detection using pre-trained VGG16 with RGB and NIR information fusion. IEEE Access, 8: 2327-2336. https://doi.org/10.1109/access.2019.2962513
[26] Phadikar, S., Sil, J. (2008). Rice disease identification using pattern recognition techniques. In 2008 11th International Conference on Computer and Information Technology, Khulna, Bangladesh, pp. 420-423. https://doi.org/10.1109/ICCITECHN.2008.4803079
[27] Hallau, L., Neumann, M., Klatt, B., Kleinhenz, B., Klein, T., Kuhn, C., Oerke, E.C. (2018). Automated identification of sugar beet diseases using smartphones. Plant Pathology, 67(2): 399-410. https://doi.org/10.1111/ppa.12741