# Corn Disease Classification Using Transfer Learning and Convolutional Neural Network

Faisal Dharma Adhinata[1], Gita Fadila Fitriana[2], Aditya Wijayanto[3], Muhammad Pajar Kharisma Putra[4]

[1,2,3]*Faculty of Informatics, Institut Teknologi Telkom Purwokerto, Indonesia*
[4]*Faculty of Engineering and Computer Science, Universitas Teknokrat Indonesia, Indonesia*

[1]faisal@ittelkom-pwt.ac.id, [2]gita@ittelkom-pwt.ac.id,
[3]aditya.wijayanto@ittelkom-pwt.ac.id, [4]pajarkharisma@teknokrat.ac.id

**Abstract - Indonesia is an agricultural country with abundant agricultural products. One of the crops used as a staple food for Indonesians is corn. This corn plant must be protected from diseases so that the quality of corn harvest can be optimal. Early detection of disease in corn plants is needed so that farmers can provide treatment quickly and precisely. Previous research used machine learning techniques to solve this problem. The results of the previous research were not optimal because the amount of data used was slightly and less varied. Therefore, we propose a technique that can process lots and varied data, hoping that the resulting system is more accurate than the previous research. This research uses transfer learning techniques as feature extraction combined with Convolutional Neural Network as a classification. We analysed the combination of DenseNet201 with a Flatten or Global Average Pooling layer. The experimental results show that the accuracy produced by the combination of DenseNet201 with the Global Average Pooling layer is better than DenseNet201 with Flatten layer. The accuracy obtained is 93% which proves the proposed system is more accurate than previous studies.**

**Keywords: Convolutional Neural Network, corn plant, DenseNet201, flatten layer, global average pooling layer**

## I. INTRODUCTION

Indonesia is an agricultural country that has extensive agricultural land and very abundant agricultural products [1]. One of the agricultural products in Indonesia, which is a staple food crop other than rice, is corn [2]. Even corn occupies the second position after rice. However, in various regions, many corn plants are attacked by diseases caused by microorganisms. The presence of this disease in corn plants can reduce the production of corn crop yields in Indonesia. Many farmers are not aware of this disease, so the farmers are late in providing treatment. One of the detections of corn disease can be seen from the leaves. This image of corn leaves is the focus our research for disease detection in corn.

Researchers processing image data usually use machine learning [3] or deep learning [4] techniques. Using this machine learning technique, the results are less optimal [3] because the data used in the training process is only 50 in each class. The highest validation results obtained are 85%. Therefore, detecting corn disease through leaf imagery requires another technique hoping that the results will be more accurate. Deep learning is a branch of machine learning techniques developed from Artificial Neural Networks [5]. The use of deep learning on image data can be used for the feature extraction process and classification [6]. Deep learning techniques on image data usually use Convolutional Neural Network (CNN). The training process uses a backpropagation-based CNN algorithm to update each iteration's weight and bias values [7].

The application of deep learning recently uses transfer learning as feature extraction and classification using CNN [8-9]. The transfer learning model is frozen that only use in the feature extraction process. Some transfer learning techniques that are quite accurate include DenseNet201 [10]. The accuracy obtained is 99% when used for the masked and unmasked face training process. The DenseNet201 model is used for the classification of two classes, namely masked and non-masked faces. Due to the achievement of DenseNet201, which is quite good, this research will use the DenseNet201 model as feature extraction.

Then, there are various ways to combine transfer learning with CNN, for example, with Flatter or Global Average Pooling as a classification layer. These two architectures will produce different models and accuracy. This research aims to get the best accuracy from the combination of the transfer learning model and CNN. The best model will be used for testing data. We use the transfer learning model as feature extraction because the initial performance achievement of the transferred knowledge is better than the performance when not using transfer learning. In addition, the training process is faster than traditional CNN [11-12].

## II. METHOD

In this research, transfer learning is used as feature extraction and CNN as a classification (Fig. 1). Before the feature extraction process is carried out, the data is resized so that the size of all input data is uniform. We use DenseNet201 or Dense Convolutional Network pre-trained model that connects each layer to other layers in a feed-forward manner [13]. DenseNet201 model is used as feature extraction of diseased corn leaf image. We use DenseNet201 as the base layer and freeze it to be first in the training stage. After the base layer, we apply a Flatten or Global Average Pooling as a classification layer.

### *A. Dataset*

This research uses corn or maize leaf disease data using PlantVillage and PlantDoc datasets [14-15]. The dataset used consists of 4 classes, namely common rust, gray leaf spot, blight, and healthy. The amount of data used in this research is common rust 1306 images, 574 gray leaf spot images, 1146 blight images, and 1162 healthy images. All processed images are RGB (Red, Green, Blue). Fig. 2 shows an example of the corn disease dataset used in this research.

### *B. Pre-processing Data*

The pre-processing stage is preparing the data before it is processed into the feature extraction and classification stage. Before being processed into feature extraction and classification stage, the dataset is resized to size of 150 x150 so that the size of datasets is uniform. The resized data is divided into 80% as training data and 20% as testing data. The distribution of training and testing data is presented in Table I.

### *C. Convolutional Neural Network*

The Convolutional Neural Network (CNN) is constructed using a series of convolutional (and occasionally pooling) layers that construct feature maps representing various features of the input images [16]. Finally, the fully connected layer or flatten layer "evaluates" these feature maps' output and generates prediction categories. However, as is the case with a great deal in the rapidly developing field of deep learning research, this strategy is being phased out in favor of a Global Average Pooling (GAP) technique. Flatten Layer converts any tensor to a one-dimensional tensor while preserving all of the tensor's values. For instance, a tensor (samples, 10, 10, 32) will be flattened to (samples, 10 * 10 * 32). This type of design increases the chances of the training dataset overfitting. Dropout layers are used to avoid overfitting. Global Average Pooling is a novel concept. It averages the spatial dimensions until they are all equal to one but keeps the other dimensions unaffected. The output of a tensor (samples, 10, 10, 32) would be as (samples, 1, 1, 32). In this research, we will examine the Flatten layer and the Global Average Pooling layer. In this research, DenseNet201 was used as a feature extraction of pre-processed data. Then we compare the Flatten layer and the Global Average Pooling layer as classification layers.

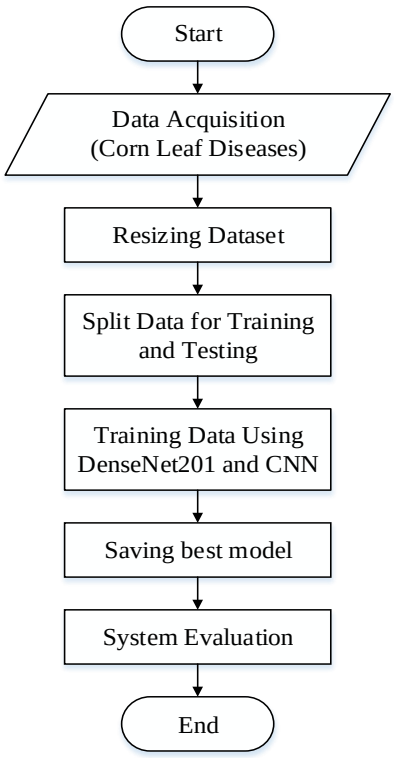
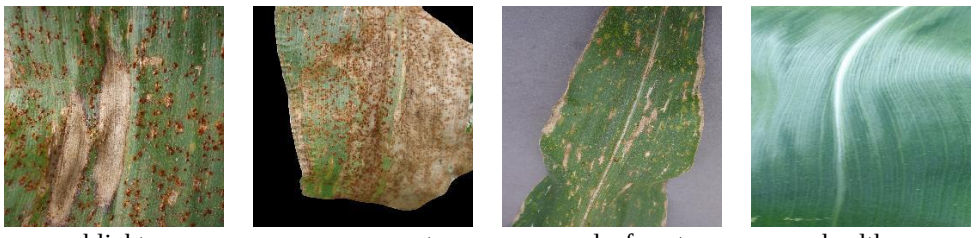



**Fig. 1 Corn disease detection system architecture**



blight common rust gray leaf spot healthy

**Fig. 2 Example of disease data on corn leaves**

TABLE I
THE DISTRIBUTION OF TRAINING AND TESTING DATA

| Data | Amount of data | Training data | Testing data |
|---|---|---|---|
| common rust | 1306 | 1045 | 261 |
| gray leaf spot | 574 | 459 | 115 |
| blight | 1146 | 917 | 229 |
| healthy | 1162 | 930 | 232 |

### *D. DenseNet201*

CNN's traditional introduction is found in [17]. However, in a traditional CNN, all layers are gradually connected, making it difficult for the network to grow deeper and wider, as it may have problems with either exploding or gradient vanishing, as in Fig. 3.

DenseNet [13] supports concatenation of all previous layer feature maps, which means that all feature maps propagate to subsequent layers and are coupled to freshly created feature maps. DenseNet's newly developed version has several advantages, including feature reuse and a reduction in the problem of either exploding or gradient vanishing. However, to make DenseNet's structure practical, the following modifications should be performed, including downsampling the feature maps to enable concatenation. If the size of the feature maps changes over time, the concatenation procedure becomes hard to perform. Then, the concept of thick blocks was developed as a means of achieving downsampling. Transition layers exist between dense blocks and comprise batch normalization, convolution, and pooling procedures. Meanwhile, Figure 3 illustrates an example of a Dense Block with a layer count of 5 and a growth rate of k. Each layer receives feature maps from the ones preceding it.

### *E. System Evaluation*

In this research, we perform a tuning parameter on both models, using Flatten and Global Average Polling layers. The parameters to be analyzed are the batch size and the number of epochs. The best batch size value will be used for the epoch number experiment. We make observations on the value of loss and accuracy. In the final stage, we evaluate using a confusion matrix to see the recall, precision, F-score, and accuracy results. The recall, precision, F-score, and accuracy formulas are shown in equations (1), (2), (3), and (4) [18].

$$Recall = \frac{TP}{TP+FN} \quad (1)$$

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

$$F - score = 2\,x\frac{recall\ x\ precision}{recall+precision} \quad (3)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FN+FP} \quad (4)$$

## III. RESULTS AND DISCUSSION

This research shows the results that we obtained from experiments using a combination of DenseNet201 feature extraction with the Global Average Polling layer or the Flatten layer. The architecture of this research is shown in Figure 4. We analyzed the training results to determine the effect of tuning the batch size and number of epoch parameters. The hardware used in this research is Core i3-9100F 4 CPU with 8 GB RAM. Configure Anaconda Python 3 software using CPU mode.

### *A. Training Result*

The training stage is the stage to get the best model on corn disease classification. This training process has been pre-arranged using the adam optimizer, then the loss function using categorical_crossentropy. The use of categorical_crossentropy is because there are four classes in this classification: common rust, gray leaf spot, blight, and healthy. Then the number of epochs used in this experiment is ten epochs. The first experiment in this research used a hyperparameter batch size. Batch size is a hyperparameter that defines the number of samples to work with before updating the internal model parameters. In a neural network, the dataset cannot pass through the entire neural network simultaneously, so it is necessary to divide the dataset into a number of datasets. This research shows experimental batch sizes with sizes 4, 8, 16, and 32 batches in Table II.

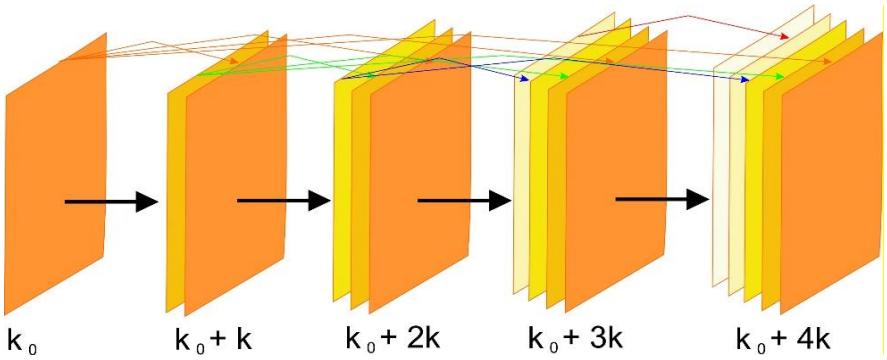



**Fig. 3 Dense block structure**

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet201 (Functional)     (None, 4, 4, 1920)        18321984
_________________________________________________________________
global_average_pooling2d (Gl (None, 1920)              0
_________________________________________________________________
dense (Dense)                (None, 4)                 7684
=================================================================
Total params: 18,329,668
Trainable params: 7,684
Non-trainable params: 18,321,984
_________________________________________________________________
```

(a)

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet201 (Functional)     (None, 4, 4, 1920)        18321984
_________________________________________________________________
flatten (Flatten)            (None, 30720)             0
_________________________________________________________________
dense (Dense)                (None, 4)                 122884
=================================================================
Total params: 18,444,868
Trainable params: 122,884
Non-trainable params: 18,321,984
_________________________________________________________________
```

(b)

**Fig. 4 Architecture in this research, (a) DenseNet201 with Global Average Pooling layer, (b) DenseNet201 with Flatten layer**

TABLE II
EXPERIMENT ON HYPERPARAMETER OF BATCH SIZE

| Model CNN | 4 batch | | 8 batch | | 16 batch | | 32 batch | |
|---|---|---|---|---|---|---|---|---|
| | loss | Acc. | loss | Acc. | loss | Acc. | loss | Acc. |
| DenseNet201 + Flatten | 2.299 | 0.917 | 1.354 | 0.921 | 0.979 | 0.911 | 0.493 | 0.92 |
| DenseNet201 + GAP | 0.258 | 0.926 | 0.223 | 0.924 | 0.215 | 0.916 | 0.194 | 0.935 |

Based on Table II, batch size 4 produces loss and accuracy values that are not optimal. The use of batch size 4 means that the first 4 data will be trained in the neural network. This too small batch size value makes the data variation a little, which causes the training results to be not optimal. The higher the batch size value, the higher the loss and accuracy values. The use of this batch size affects the resulting loss and accuracy values. The loss value will continue to decrease as the batch size increases. However, on the accuracy value, through 4

batch size trials, the best results were obtained with a batch size value of 32, which indicates that this dataset is the most suitable for using 32 inputs to be passed to the neural network. In this experiment, the optimal result was obtained at a batch size of 32. Therefore, in the hyperparameter experiment, the number of epochs will use a batch size of 32. We will evaluate the number of epochs in multiples of 5 from 5 to 50. Table III shows the number of epochs on the DenseNet201 model with a Flatten Layer or Global Average Pooling layer. Loss is training loss, which is the value of calculating the loss function from the training dataset and predictions from the model. Accuracy is training accuracy, which is the value of calculating the accuracy of the training dataset and predictions from the model. Validation loss is the calculation value of the loss function from the validation dataset and predictions from the model with input data from the validation dataset. Validation accuracy is the value of calculating the accuracy of the validation dataset and predictions from the model with input data from the validation dataset.

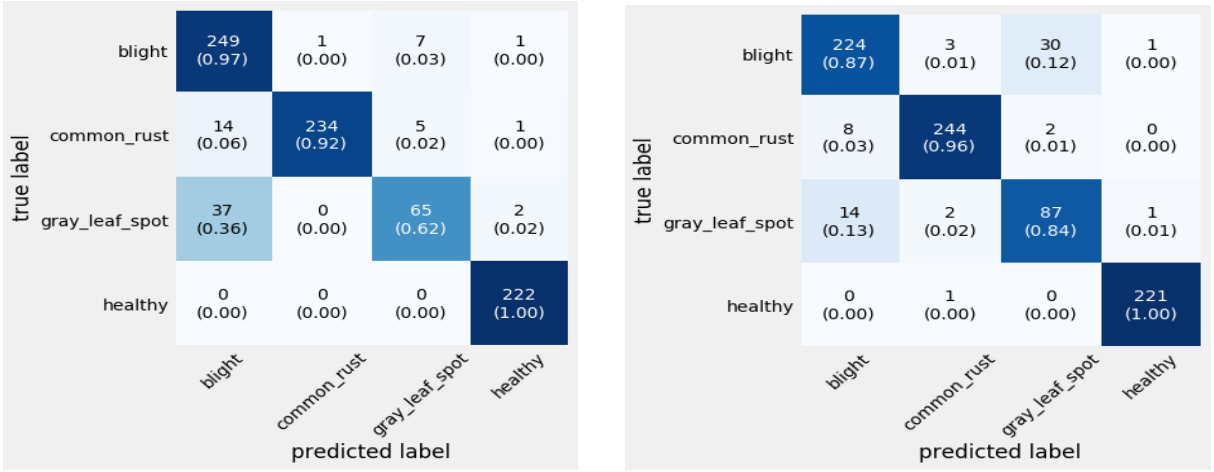
Based on Table III, the optimal result for the DenseNet201 training model with Flatten Layer or Global Average Pooling Layer is at epoch 35. In this testing stage, our evaluation benchmark is the validation accuracy value. The use of validation accuracy is because we evaluate the accuracy value of the validation results as much as 20% in the dataset. It appears that the overall value of validation accuracy using the combination of DenseNet201 with Global Average Pooling Layer is better than the combination of DenseNet201 with Flatten Layer. This best model will be used for testing, which will be evaluated using a confusion matrix.

### *B. Testing Result and Analysis*

In this research, the ModelCheckpoint function stores the results based on the best validation accuracy value. The best model is used for evaluation using a confusion matrix. Fig. 5 shows the prediction results of this research, while Table IV shows the combination DenseNet201 with the Global Average Pooling Layer.

TABLE III
EXPERIMENT ON HYPERPARAMETER OF THE NUMBER OF EPOCHS

| Epoch | DenseNet201 + Flatten | | | | DenseNet201 + GAP | | | |
|---|---|---|---|---|---|---|---|---|
| | loss | Acc. | Val. loss | Val. Acc. | loss | Acc. | Val. loss | Val. Acc. |
| 5 | 0.0440 | 0.9863 | 0.4895 | 0.9033 | 0.1549 | 0.9419 | 0.2217 | 0.9260 |
| 10 | 0.0648 | 0.9816 | 0.5463 | 0.9177 | 0.0989 | 0.9698 | 0.2328 | 0.9200 |
| 15 | 0.0353 | 0.9910 | 0.7824 | 0.9093 | 0.0775 | 0.9801 | 0.2248 | 0.9260 |
| 20 | 0.0091 | 0.9973 | 0.8328 | 0.9141 | 0.0603 | 0.9855 | 0.2228 | 0.9260 |
| 25 | 0.1237 | 0.9847 | 0.9468 | 0.9177 | 0.0489 | 0.9930 | 0.2337 | 0.9272 |
| 30 | 0.0143 | 0.9955 | 1.3057 | 0.9093 | 0.0488 | 0.9904 | 0.2469 | 0.9260 |
| 35 | 0.0110 | 0.9992 | 1.0074 | 0.9248 | 0.0358 | 0.9933 | 0.2600 | 0.9308 |
| 40 | 0.0663 | 0.9932 | 1.6252 | 0.9069 | 0.0290 | 0.9964 | 0.2662 | 0.9260 |
| 45 | 0.0245 | 0.9977 | 1.4654 | 0.9177 | 0.0250 | 0.9967 | 0.2836 | 0.9236 |
| 50 | 0.0922 | 0.9905 | 1.7370 | 0.9189 | 0.0233 | 0.9965 | 0.3048 | 0.9260 |




(a) (b)

**Fig. 5 Confusion matrix results from DenseNet201 combination with (a) Flatten layer, (b) Global Average Pooling Layer**

TABLE IV
TESTING RESULTS ON DENSENET201 MODEL WITH FLATTEN OR GLOBAL AVERAGE POOLING LAYER

| | DenseNet201 + Flatten | DenseNet201 + GAP |
|---|---|---|
| **Recall** | 0.8775 | 0,9175 |
| **Precision** | 0.9125 | 0,9025 |
| **F-measure** | 0.89 | 0.9075 |
| **Accuracy** | 0.92 | 0.93 |

Table IV shows the combination DenseNet201 with the Global Average Pooling Layer better than with the Flatten layer. This value is influenced by the recall results, as shown in Fig. 5. The type of gray leaf spot disease produces the lowest recall value compared to other types of diseases. The low recall value is because the amount of data on gray leaf spot disease is also the smallest. However, the Global Average Pooling layer classification still produces a high recall value, which is 0.84. This result is also inseparable from the advantages of using the Global Average Pooling layer, which can reduce overfitting by reducing the number of parameters, as shown in Fig. 3. The total results of using the Global Average Pooling layer are less than the Flatten layer. Table IV also shows that the DenseNet model with the Global Average Pooling layer is not overfitting because the loss and accuracy values are not too far apart, different from what is produced in the Flatten layer. Overall, the results of this research are better than previous studies [3], namely, the highest validation accuracy is only 85%. These results prove that the method proposed in this research is more accurate than previous research using machine learning. An example of the prediction results in this research is shown in Fig. 6.

As seen in Fig. 6 shows that the actual and predicted results are the same. Number 0 indicates common rust corn disease, number 1 indicates gray leaf spot corn disease, number 2 indicates corn blight disease, and number 3 indicates healthy corn. These results show that the corn disease classification process using a leaf cross-section in this system is quite accurate. These results can provide recommendations for farmers for early detection of corn plant diseases so that treatment can be carried out as soon as possible.

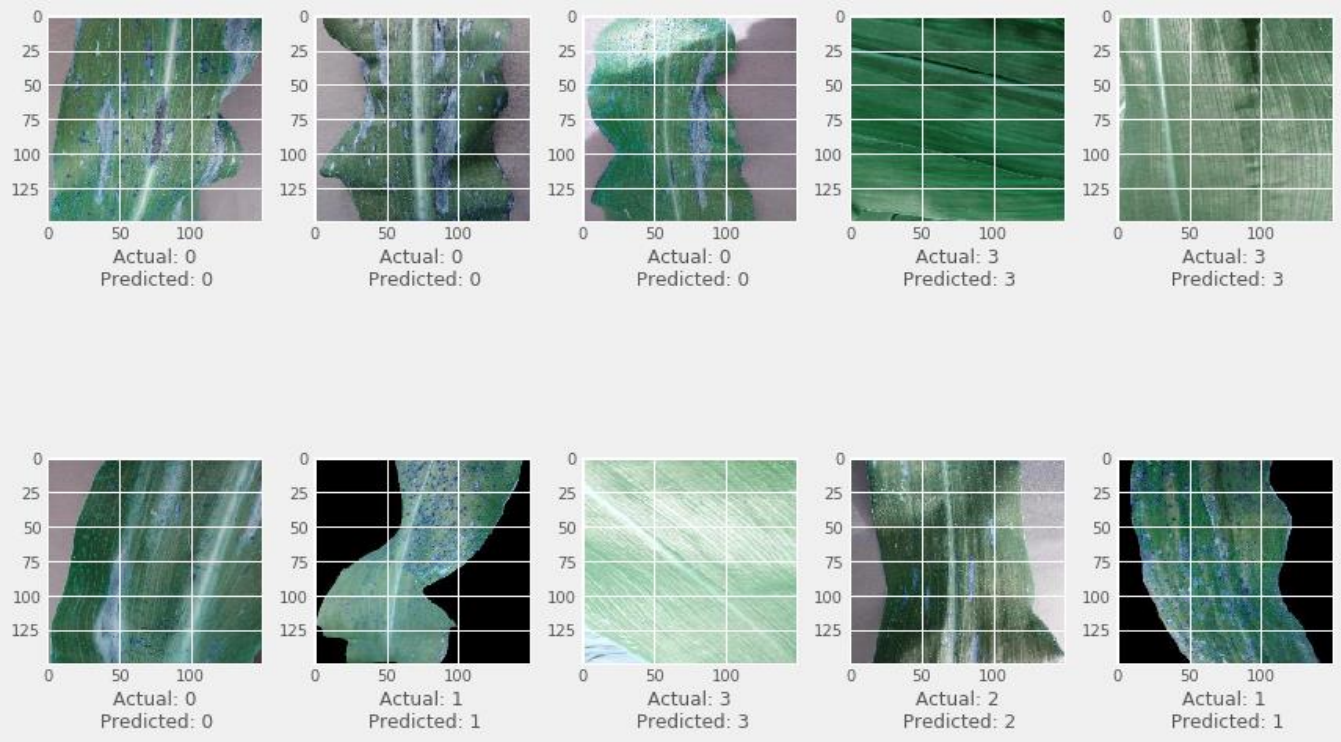



**Fig. 6 Prediction results of corn disease detection**

## IV. CONCLUSION

Diseases in plants, especially corn plants, must be identified early to get treatment immediately to obtain optimal yields. Previous research uses machine learning techniques that resulted in less optimal results so that disease predictions did not match the actual disease. This research using deep learning techniques with the hope that the accuracy value will increase. We use a combination of transfer learning as feature extraction and CNN as classification. Optimal results were obtained using the DenseNet201 model with the Global Average Pooling layer with 93% accuracy. In this research, there are still have limitations. The dataset used is imbalanced, which makes the recall value not optimal. Researchers can combine feature extraction techniques and other classifications to obtain higher accuracy on this imbalanced data in future research.

## REFERENCES


[1] D. Pitaloka, "Hortikultura: Potensi, Pengembangan Dan Tantangan," *Jurnal Teknologi Terapan: G-Tech*, vol. 1, no. 1, pp. 1–4, 2020.

[2] R. P. Ramadhan and N. L. Marpaung, "Identifikasi jenis penyakit daun tanaman jagung menggunakan jaringan saraf tiruan berbasis backpropagation," *Jom FTEKNIK*, vol. 6, no. 1, pp. 1–5, 2019.

[3] W. Setiawan, M. Syarief, and N. Prastiti, "Maize Leaf Disease Image Classification Using Bag of Features," *Jurnal Infotel*, vol. 11, no. 2, pp. 48–54, 2019.

[4] M. Syarief and W. Setiawan, "Convolutional neural network for maize leaf disease image classification," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 3, pp. 1376–1381, 2020.

[5] A. Hidayat, U. Darusalam, and I. Irmawati, "Detection of Disease on Corn Plants Using Convolutional Neural Network Methods," *Jurnal Ilmu Komputer dan Informasi*, vol. 12, no. 1, pp. 51, 2019.

[6] M. Dyrmann, H. Karstoft, and H. S. Midtiby, "Plant species classification using deep convolutional neural network," *Biosystems Engineering*, vol. 151, pp. 72–80, 2016.

[7] O. Sudana, I. W. Gunaya, and I. K. G. D. Putra, "Handwriting identification using deep convolutional neural network method," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 18, no. 4, pp. 1934–1941, 2020.

[8] R. Prathivi, "Optimasi Model TL-CNN Untuk Klasifikasi Citra CIFAR-10," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 1, no. 10, pp. 3–7, 2021.

[9] Stephen, Raymond, and H. Santoso, "Aplikasi Convolutional Neural Network untuk Mendeteksi Jenis-jenis Sampah," *Explore: Jurnal Sistem informasi dan telematika (Telekomunikasi, Multimedia dan Informatika)*, vol. 10, no. 2, pp. 122–130, 2019.

[10] F. D. Adhinata, D. P. Rakhmadani, M. Wibowo, and A. Jayadi, "A Deep Learning Using DenseNet201 to Detect Masked or Non-masked Face," *JUITA: Jurnal Informatika*, vol. 9, no. 1, pp. 115–121, 2021.

[11] D. Han, Q. Liu, and W. Fan, "A new image classification method using CNN transfer learning and web data augmentation," *Expert Systems with Applications*, vol. 95, pp. 43–56, 2018.

[12] S. Giri and B. Joshi, "Transfer Learning Based Image Visualization Using CNN," *International Journal of Artificial Intelligence & Applications*, vol. 10, no. 4, pp. 47–55, 2019.

[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 2261–2269, 2017.

[14] D. Singh, N. Jain, P. Jain, P. Kayal, S. Kumawat, and N. Batra, "PlantDoc: A dataset for visual plant disease detection," *ACM International Conference Proceeding Series*, February, pp. 249–253, 2020.

[15] G. G. and A. P. J., "Identification of plant leaf diseases using a nine-layer deep convolutional neural network," *Computers & Electrical Engineering*, vol. 76, pp. 323–338, 2019.

[16] A. Patil and M. Rane, "Convolutional Neural Networks: An Overview and Its Applications in Pattern Recognition," *Smart Innovation, Systems and Technologies*, vol. 195, pp. 21–30, 2021.

[17] Y.-D. Zhang, C. Pan, X. Chen, and F. Wang, "Abnormal breast identification by nine-layer convolutional neural network with parametric rectified linear unit and rank-based stochastic pooling," *Journal of Computational Science*, vol. 27, pp. 57–68, 2018.

[18] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.