Available online at :
http://ejournal.amikompurwokerto.ac.id/index.php/telematika/

Telematika

Accreditated SINTA "2" Kemenristek/BRIN, No. 85/M/KPT/2020

UNIVERSITAS AMIKOM PURWOKERTO

CNN Pruning for Edge Computing-Based Corn Disease Detection with a Novel NG-Mean Accuracy Loss Optimization

Aji Gautama Putrada[1], Ikke Dian Oktaviani[2], Mohamad Nurkamal Fauzan[3] , Nur Alamsyah[4]

[1.2, 3, 4] School of Computing,
[1.2, 3, 4] Telkom University, Bandung, Indonesia
E-mail:ajigps@telkomuniversity.ac.id[1], oktavianiid@telkomuniversity.ac.id[2], mnurkamalfauzan@student.telkomuniversity.ac.id[3], nuralamsyah@student.telkomuniversity.ac.id[4]

ARTICLE INFO

History of the article:
Received April 18, 2024
Revised August 02, 2024
Accepted August 05, 2024

Keywords:
Convolutional Neural Network
Pruning
Edge Computing
Corn Disease Detection
Accuracy Loss Optimization

Correspondece:
E-mail:
oktavianiid@telkomuniversity.ac.id

ABSTRACT

Plant disease detection studies disease attacks in plants detected on the leaves using computer vision. However, some plant disease detection solutions still utilize cloud computing, where the problems include slow processing times and misuse of data privacy. This study aims to evaluate the performance of convolutional neural network (CNN) pruning in edge computing-based plant disease detection. We use Kaggle's plant disease image dataset, which contains three corn diseases. We also created an edge computing system architecture for plant disease detection utilizing the latest communication technology and middleware. Next, we developed an optimal CNN model for plant disease detection using grid search. We pruned the CNN model in the final step and tested its performance. In this step, we developed a novel normalized-geometric mean (NG-mean) method for accuracy loss optimization. The test results show that class weights can optimize specificity and g-mean on the imbalanced dataset, with values of 0.995 and 0.983, respectively. The grid search results then optimize the optimization method's hyperparameters, learning rate, batch size, and epoch to achieve the highest accuracy of 0.947. Applying pruning produces several models with variations in sparsity and scheduling methods. We used the new NG-mean method to find the best compressed model. It had constant scheduling, 0.8 sparsity, a mean accuracy loss of 1.05%, and a CR of 2.71×. This study enhances the efficiency and privacy of plant disease detection by utilizing edge computing and optimizing CNN models, leading to faster processing and better data security. Future work could explore the application of the novel NG-Mean method in other domains and the integration of additional plant species and diseases into the detection system.

INTRODUCTION

Plant disease detection is a branch of science that studies disease attacks in plants detected through observation of the plant's leaves, which can be done directly, through the lab, or by computer vision (Yucky et al., 2021). Blight, common rust, and gray leaf spots are a plant disease that attacks plant tissue and is caused by a fungus, which can cause crop failure (Anim-Ayeko et al., 2023). Several studies in the field of artificial intelligence (AI) has been used to tackle crop failure, for example a research (Dagwale & Adakane, 2023) that can detect several types of plants and diseases with high accuracy. Deep learning methods, including convolutional neural networks (CNN), can detect blight on leaves of plants with good performance (Al-Adhaileh et al., 2023). However, some plant disease detection solutions still detect in the cloud, where the image is sent to the internet first (Shrestha et al., 2020). Problems with using the cloud include slow processing times and misuse of data privacy (Prabowo et al., 2023).

Several studies have overcome this problem with edge computing, where data processing occurs on the device when the image is captured (Putrada, Abdurohman, et al., 2023a). Some novel edge architectures have been provided, for example, FedEdge (Ye et al., 2020), which targets edge computing to lighten the burden of mobile device computing and network communication. CREAT (Cui et al., 2022) is edge computing that adds up blockchain to the system to enhance the security and privacy while exercising a decentralized architecture. CREAT provides predictive caching to increase the hit rate of each edge node in the system. The novelty of LoPECS (Tang et al., 2020) lies in a heterogenous resource scanning system that optimizes edge device battery efficiency on autonomous vehicles. The cloudlet provided by LoPECS is developed using an Nvidia Jetson TX1.

However, running a CNN on an edge device requires a pruning process because the resources on the edge device are not powerful enough to run complex algorithms, where pruning can reduce the number of weights in the CNN model (Joardar et al., 2023). CroApp (Jia et al., 2022) is a compression model for CNN, where pruning is one of the methods. The optimal CroApp model can provide a compression ratio (CR) of 1.81x while increasing the accuracy of the CNN model in making predictions. Another study (Moon et al., 2019) succeeded in applying pruning to CNN and obtained a compression ratio of up to 3.00x. ThiNet (Luo et al., 2018) is a CNN pruning method targeted for thin deep learning in mobile phones and embedded gadgets. ThiNet can compress the VGG-16 pre-trained model to only 2.66 MB of model size. Applying CNN pruning to plant disease detection is a research opportunity.

The research aim of this study is to evaluate the performance of CNN pruning in edge computing-based plant disease detection. We use the plant disease image dataset from Kaggle, where the data contains three diseases in corn. We also created an edge computing system architecture for plant disease detection utilizing the latest communication technology and middleware. Next, we developed an optimal CNN model for plant disease detection using grid search. In the final step, we pruned the CNN model and tested its performance. In this step, we developed a novel normalized-geometric mean (NG-Mean) method for accuracy loss optimization.

To the best of our knowledge, there has never been any research that applies CNN pruning to edge computing-based plant disease detection. The following is a list of contributions to this research. First is an architecture for deep learning-based corn disease detection that applies edge computing and also pruning for model compression. Secondly, we propose a corn disease detection that uses class weights with optimum specificity and g-mean on the imbalanced dataset. Third is the grid search optimization method for corn leaf disease detection using CNN, which can find the optimal optimization method, learning rate, batch size, and epoch. Fourth is a novel method called NG-Mean for accuracy loss optimization in deep learning pruning.

The systematics of this paper for the remaining sections are as follows: Section 2 shows the flow of our research and the theories involved in each stage. Section 3 presents the results of our research and also discusses them in relation to state-of-the-art research. Lastly, Section 4 provides the conclusions of our research and how we have achieved the research aim that we set out.

RESEARCH METHODS

We implement a research workflow to accomplish our research goal. We use Kaggle's plant disease image dataset, which contains three corn diseases. We also created an edge computing system architecture for plant disease detection utilizing the latest communication technology and middleware. Next, we developed an optimal CNN model for plant disease detection using grid search. We pruned the CNN model in the final step and tested its performance. In this step, we developed a novel method called NG-Mean for accuracy loss optimization.

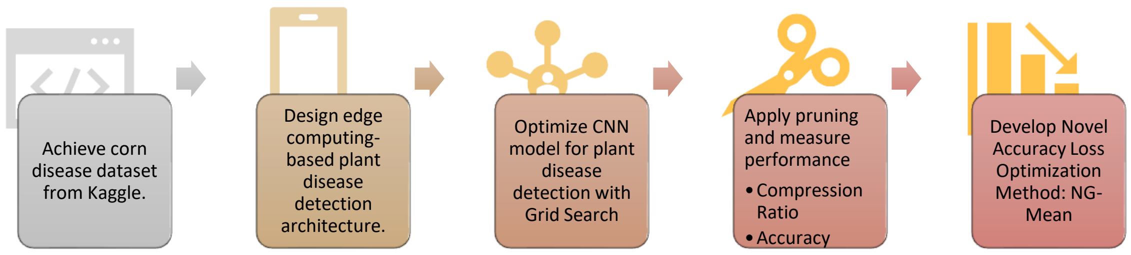
Achieve corn disease dataset from Kaggle.
Design edge computing-based plant disease detection architecture.
Optimize CNN model for plant disease detection with Grid Search
Apply pruning and measure performance
• Compression Ratio
• Accuracy
Develop Novel Accuracy Loss Optimization Method: NG-Mean

Figure 1. Our Proposed Research Work Flow.

1. Corn disease detection

Plant disease detection in computer vision is one implementation that detects plant disease attacks by observing leaf images (Velmurugan et al., 2023). Several diseases can occur in corn plants, which can be detected on the leaves, namely blight, common rust, and gray leaf spot (Ashwini & Sellam, 2023). Fungi cause these diseases, and if a crop is affected by this disease, it can cause lesions, wilting, and decreased yield. If blight, common rust, or gray leaf spot is detected on a corn plant, to avoid the consequences, farmers must isolate it, improve the quality of life of the corn, and apply fungicide to the plant. These fungicides include azoxystrobin and triazoles (Degani et al., 2018).

We obtained the corn leaf image dataset from Kaggle, where the dataset was uploaded by Smaranjit Ghose (Pamungkas et al., 2023). The dataset consists of 4188 corn leaf images. There are four classes: normal leaf, blight leaf, common rust leaf, and gray leaf spot leaf. The uploader combines two well-known datasets related to corn leaf, namely PlantVillage and PlantDoc. Some of the four related labels can be combined, which is done to study label correlation and overcome the problem of dataset imbalance (Siahroudi & Kudenko, 2023). Figure 1 shows four samples from each leaf condition.

In addition to the Kaggle dataset uploaded by Smaranjit Ghose, the PlantVillage dataset, which contains images of various plant diseases across different species, and the AI Challenger dataset, which focuses on agricultural disease images from China, are several other datasets available for plant disease detection. We chose the Kaggle dataset by Smaranjit Ghose specifically because it is well-curated, widely recognized, and focused exclusively on corn, aligning perfectly with the scope of our research. This dataset provides high-quality images of corn leaves with three distinct diseases, ensuring a targeted and comprehensive study. Furthermore, the dataset's structure and annotations are suitable for training and evaluating convolutional neural network (CNN) models, making it an ideal choice for our edge computing-based approach.

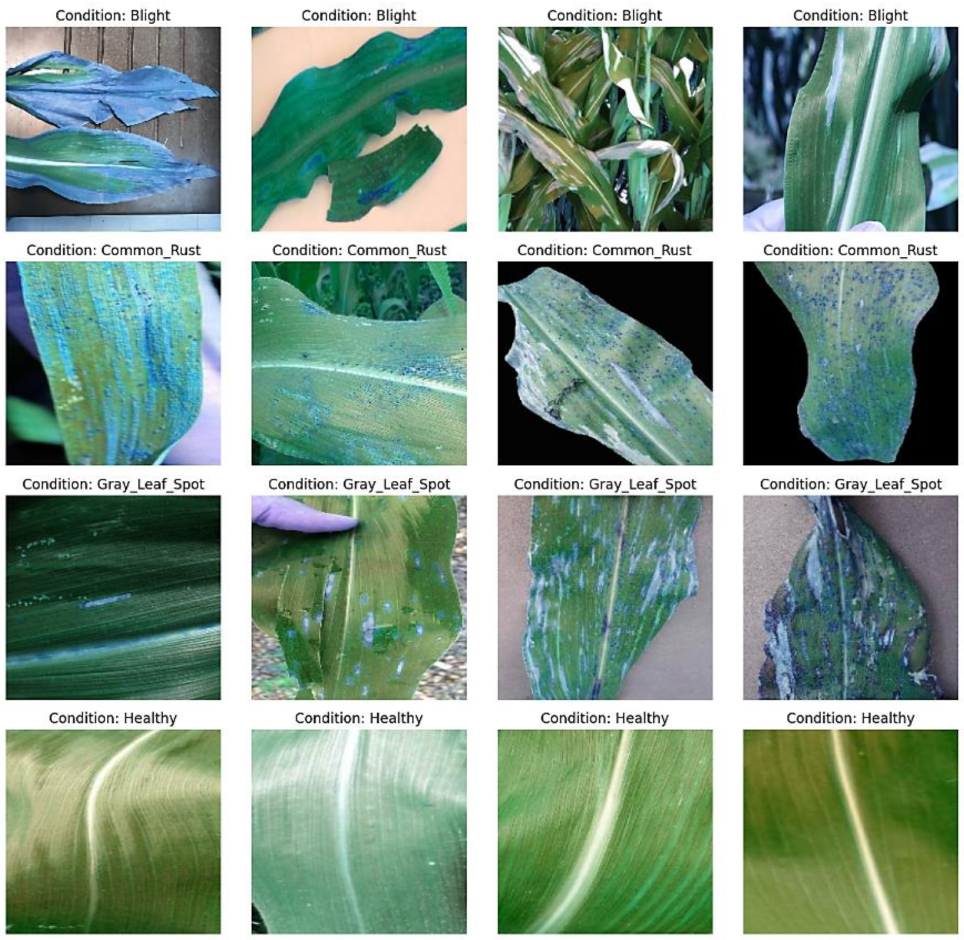

Figure 1. Leaf Dataset Samples.

The dataset goes through several deep learning pre-processing techniques before entering the prediction stage. Image resizing is a useful step to adjust the image size to the input size of the deep learning model. Normalization changes the values in the image to have a standard range (Vocaturo et al., 2018). The label encoder functions to change the text label in the categorical label into a numerical label. One hot encoder function to transform numerical labels into binary vectors, ensuring that the output in the SoftMax activation function output has a sum of 1 (Pandey & Dukkipati, 2017).

2. CNN model training

Deep learning is an advancement of the science of machine learning where the learning model can carry out layered learning, where the first layer of learning is feature learning (Putrada, Alamsyah, et al., 2023). CNN is a type of deep learning where feature learning is carried out by applying convolutional kernels. These kernels perform convolution operations on images with a certain matrix size to extract features that are useful for predictions in the next layer (Pane et al., 2022).

Deep learning with one CNN layer is sometimes referred to as shallow CNN, where several studies highlight the advantages of the shallow CNN architecture. A paper (Lei et al., 2020) states that the benefits of shallow learning are shorter training time, better accuracy, and low space and time complexity. The CNN architecture that we propose for corn disease detection is also a shallow CNN. Figure 2 shows the shallow CNN architecture that we suggest. The corn leaf image that is input is the result of the resize to 28×28. This corn leaf image has an RGB color model, so its dimensions are 28×28×3. The CNN architecture comprises of six layers, often referred to as the LeNet-5 architecture.

The input shape adapts to the image size which is 28×28×3. The second layer is the convolution layer which uses a kernel with the size of 3×3. The third layer is the max pooling layer with a window that has a size of 2×2. There are 36 units each in the second and third layer. The fourth layer is the flatten layer, where the fifth layer is the dense layer which consists of 256 layers. The activation function of the fifth layer is the rectified linear unit (ReLU). Lastly, the sixth layer is the output layer with SoftMax activation function and uses two units which adapt to the number of classes in the dataset.

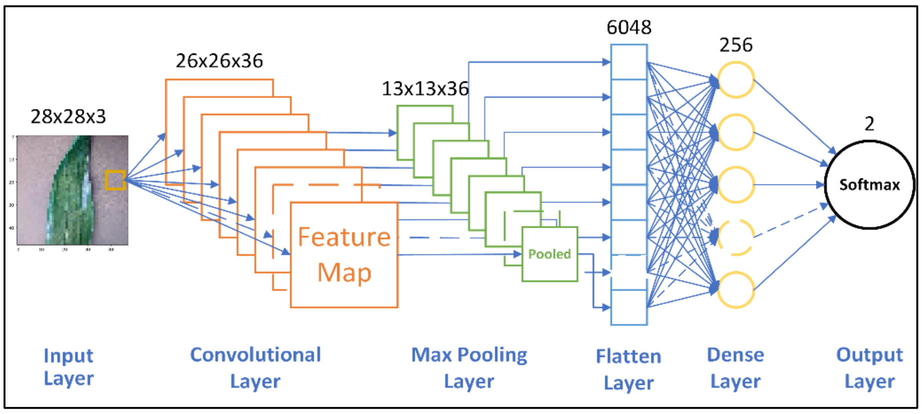

Figure 2. Our Proposed Shallow CNN for Corn Disease Detection.

Furthermore, on the convolutional layer, we apply 36 2D-CNN filters with a size of 3×3. The feature map (S_m) that is generated based on the input size (S_i) and the number of filters (S_k) is based on the following formula:

$$S_m = \frac{S_i - S_k}{Stride} + 1 \quad (1)$$

where Stride is the distance of the filter movement each time period. Based on this formula, the size of the output feature map from the convolutional layer is 26×26×36. The max pooling layer is a layer that carries out the process of reducing map features by finding the maximum value of a sliding window (Escobar et al., 2022). The number of output feature maps from the max pooling layer does not change from the input. Meanwhile, the output size of the max pooling layer (S_p) follows the following formula:

$$S_p = \frac{S_m}{S_w} \quad (2)$$

where S_w is the size of the max pooling window, and our max pooling window is 2, so our output x number of output max pooling layers is 13×13×36. The last three layers in the architecture are the flatten layer, which converts neurons into one-dimensional; the fully connected layer and the output is SoftMax with size 2.

Regarding the state of the dataset, we need to anticipate it with several optimization techniques in the CNN model. A dataset can experience imbalance, that is, if one label is much more numerous than other labels. If an imbalance occurs, then the optimization on the CNN that can be done is class weight assignment. Class weight assignment is a class weight calculation based on the occurrence frequency of each class and predicted probability (Fernando & Tsokos, 2022). Predicted probability is the likelihood that is assigned to each class during classification. The following is the formula for calculating the class weight of each class (W_c):

$$W_c = \frac{\mathrm{N}}{\mathrm{C}\times\mathrm{N_c}}, c \in C \quad (3)$$

where N is the dataset size, C is the number of classes, and N_c is the dataset size in class c.

We use accuracy, sensitivity, specificity, and g-mean to compare the performance of models with class weights and without class weights. Accuracy is a metric that compares all data predicted correctly with all data. Sensitivity in binary classification is a metric that measures the model's ability to predict the label "1". Specificity, on the other hand, is a metric that measures the model's ability to predict the label “0”. Finally, g-mean is an aggregate of sensitivity and specificity metrics calculated by taking the square root of the product of these two values.

Deep learning using CNN has so many hyperparameters, so hyperparameter tuning in CNN learning can be a challenge (Zhan, 2022). Grid search can overcome the complexity and trade-offs in deep learning, where grid search is an optimization method for hyperparameter tuning. The way grid search works is by carrying out a systematic search based on a grid of pre-defined hyperparameters, whereby carrying out this mechanism, the optimal hyperparameters can be found (Thanh, 2021). We include four hyperparameters in our grid search: learning rate (LR), epoch, batch size, and optimizer. We brought two values to compare for each hyperparameter, namely 0.01 and 0.001 for LR, 10 and 50 for epoch, 6 and 24 for batch size, and Adam and stochastic gradient descent (SGD) for the optimizer. The formula for calculating the number of outputs produced by grid search (R) based on the number of hyperparameters (H) and the number of values for each hyperparameter (v_h) is as follows:

$$R = \prod_{h=1}^{H} v_h \quad (4)$$

with this formula, the search grid that we have set has $R = 16$. Table 1 summarizes an explanation of the hyperparameters optimized in grid search.

Table 1. Hyperparameters to be Optimized in the Grid Search and their Compared Values.

No.	*Hyperparameter*	*Compared Values*
1	LR	0.01, 0.001
2	Epoch	10, 50
3	Batch size	6, 24
4	Optimizer	Adam, SGD

3. Edge computing and pruning

Edge computing is a concept that aims to improve cloud architecture by bringing the computing process closer to the end device so that processing time can be shortened for systems with real-time constraints (Putrada et al., 2024). Our system uses the corn disease detection process on a cellphone as an edge device connected to a cloud server (Shrestha et al., 2020). Instead of sending images of corn leaves to the cloud, the smartphone application processes them. In contrast, the classification results from corn disease detection are sent to the cloud for further analysis (Choudhary et al., 2022). HTTP is a middleware that can communicate between applications on edge devices and cloud services. Figure 3 shows our explanation in the form of an architectural drawing of edge computing for corn disease detection.

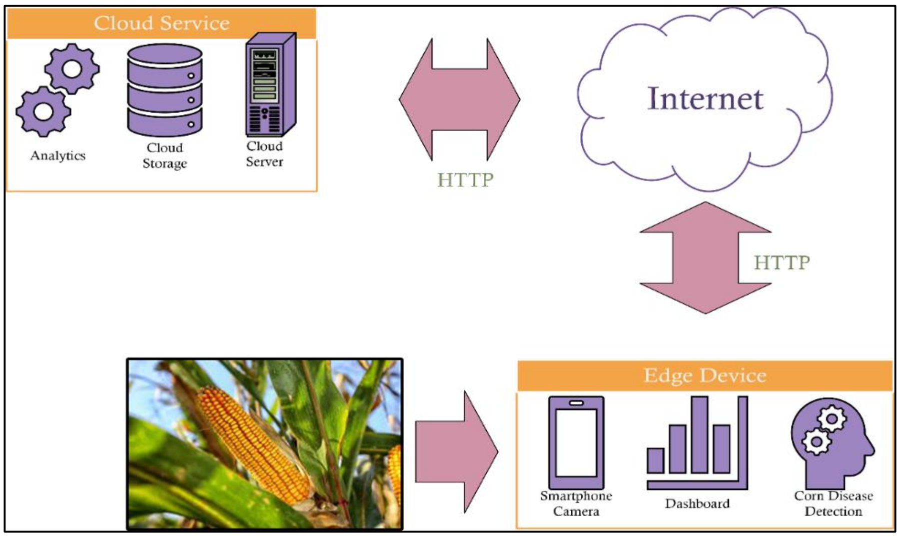

Figure 3. The Proposed Corn Disease Detection System Based on Edge Computing Architecture.

The problem in applying deep learning to edge computing is limited resources to run deep learning. With these limitations, model compression becomes an important factor. Model compression is the process of shrinking an intelligence model to better adapt to the resources on which it runs (Putrada, Abdurohman, et al., 2023b). Pruning in deep learning is a method of model compression that reduces the weights in the neural network while maintaining the prediction performance of the deep learning model. In this way, the size of the deep learning model will also be reduced (Gangopadhyay et al., 2023).

Moreover, magnitude-based pruning is a type of pruning criterion that removes small weights in deep learning models (Ahia et al., 2021). Sparsity in pruning explains what proportion of the weight will be pruned. Pruning should apply a scheduling mechanism because applying pruning during training with constant sparsity can cause overfitting or underfitting. Polynomial decay scheduling is implementing scheduling by increasing sparsity in a scheduled manner with an increase in the polynomial trend (Fan et al., 2002). The formula for polynomial decay scheduling is as follows:

$$S_e = S_0 \times \left(1 - \frac{E}{E_p}\right)^{\mathrm{D}}, e \in E \quad (5)$$

where S_e is the sparsity of the current epoch, S_0 is the initial sparsity, E is the number of epochs in deep learning training, E_p is the number of epochs for pruning, and D is the polynomial degree, which is a hyperparameter in the equation. Two measures of model compression performance are accuracy loss and CR, where the formula of CR is the original model size divided by the model size after applying compression (Qin & Sun, 2023). The formula of accuracy loss is the percentage of the accuracy decline from the original model's accuracy.

Finding an optimum compression model based on decreasing accuracy and CR is challenging (Kurtic et al., 2022). Here, we propose a novel formula to find the optimum model from several choices with variations in sparsity, scheduling, accuracy, and CR called NG-Mean:

$$NG - Mean_s = \sqrt{(1 - A'_s) \times Z'_s}, s \in S \quad (6)$$

$$s_o = argmax(NG - Mean) \quad (7)$$

where S is the range of sparsity values, A' is normalized accuracy, Z' is normalized size, and s_o is the index of the optimum model.

RESULTS AND DISCUSSION

1. Results

The corn leaf image dataset that we downloaded and processed has two labels: "Healthy" and "Not Healthy." After going through the label encoder, the labels become 1 and 0, respectively. The number of datasets with label 1 is 3,026, and label 0 is 1,162, where the ratio condition of 2.6:1 shows an imbalance in the dataset. In the first test, we observed the performance of class weight in dealing with this imbalance.

Figure 4 compares the CNN model in predicting cord disease with and without class weights. We use boxplot comparison to ensure statistical requirements for the comparison, where 50 samples are collected by testing the model iteratively. Using class weights can increase the model's mean accuracy from 0.94 to 0.96. This indicates that balancing the influence of different classes in the training process enhances the model's ability to correctly classify diseases in corn leaves. Then, the increase in mean specificity is from 0.80 to 0.87, which shows a significant increase in the capability to detect the minority label. Lastly, the increase in mean specificity has an impact on the increase in mean g-mean, which is from 0.89 to 0.93. This result shows that the capability of the class weights to increase the performance of the model to detect the minority label influences of the overal prediction capability of the disease detection model.

All the increase of mean performances are significant, as they all satisfy the t-test for statistically significant difference between the means of two groups. The p-values of the t-test for accuracy, sensitivity, specificity, and g-mean are 0.00018, 0.00284, 0.00015, and 0.00019, respectively. Because all the values are below the α, which is 0.05, it indicates that all four t-test results accept the alternative hypothesis h_1, that is, there is a significant difference between the means of each couple of groups. The highest p-value belongs to sensitivity and the lowest belongs to specificity, which shows that the class weights impact most on the much important label, the minority label.

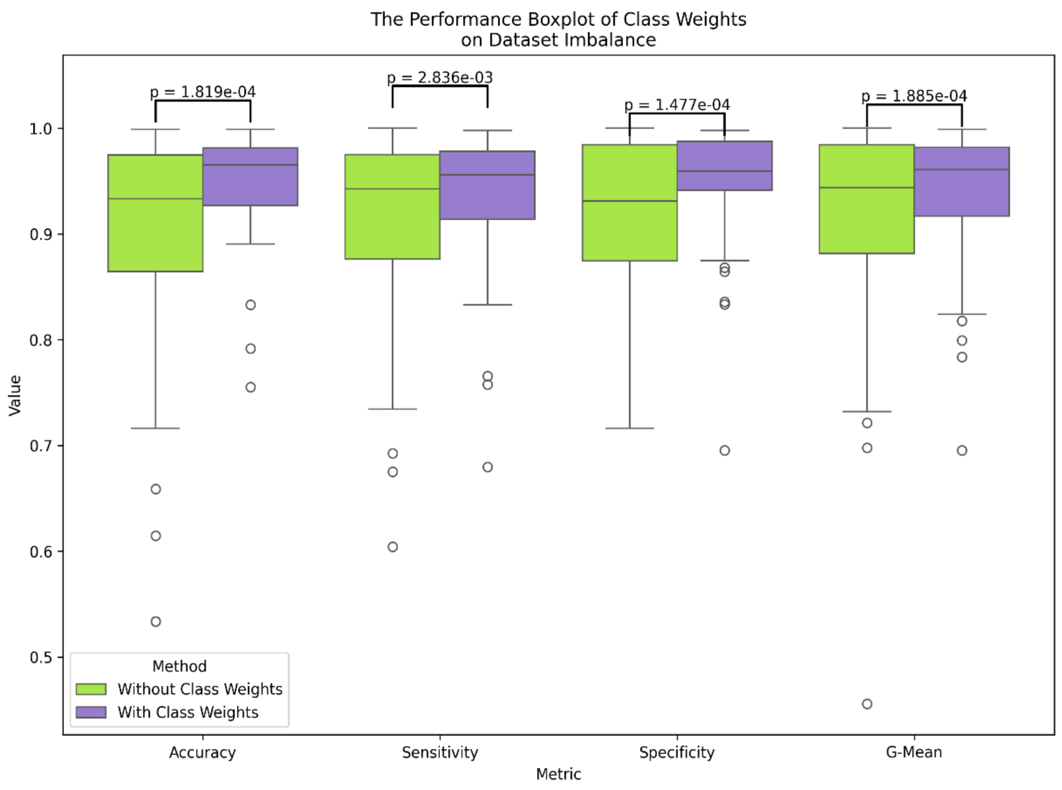

Figure 4. The Performance Comparison of the usage of Class Weights in Dealing with Dataset Imbalance.

In the following tests, we optimize the model with grid search, using four hyperparameters with two values each. Our grid search yielded 16 results, of which Table 2 shows all of these results. First, we observe that Adam is a more optimal optimizer than SGD. However, an LR of 0.001 is more optimal for Adam than an LR of 0.01. On the other hand, in SGD, LR with a value of 0.01 is more optimal than LR with a value of 0.001. Further, a batch size of 24 is more optimal than a batch size of 6. Finally, between the two epoch values, the optimum epoch value for corn disease detection is 50 versus 10. The optimum hyperparameters for the CNN model are obtained based on these observations. is the Adam optimizer with LR = 0.001, batch size = 24, and epoch = 50. The accuracy of the CNN model with these hyperparameters is 0.974.

Table 2. Accuracy Comparison of Grid Search for CNN Model Optimization Results.

Batch Size	Epoch	Adam		SGD	
		LR=0.001	LR=0.01	LR=0.001	LR=0.01
24	50	**0.974**	0.973	0.720	0.879
	10	0.918	0.858	0.720	0.722
6	50	0.864	0.750	0.732	0.820
	10	0.823	0.777	0.720	0.745

In the final test, we applied pruning, where we compared two scheduling methods of pruning: constant vs polynomial decay scheduling. We varied the sparsity value from 0.3 to 0.9 with ten times iteration to collect the distribution of accuracies. The hypothesis is that the greater the sparsity, the greater the CR. Figure 5 compares the two scheduling methods with a boxplot graph, where three points can be observed. The first point is that both constant and polynomial decay scheduling experience a decrease in mean accuracy as the sparsity value increases. The second point is that for sparsity 0.3 to 0.6, the resulting polynomial decay and constant model has no significant difference in mean accuracy. This has been tested using the t-test, where the p-values are all above the α, which is 0.05. The third observation is that for sparsity 0.7 to 0.9, the mean performance of polynomial decay is worse than constant scheduling, where the decrease in polynomial decay performance has a larger slope than constant scheduling. However, according to t-test, the difference is only significant at sparsities 0.7 and 0.8, where there are no significance in difference at sparsity equals 0.9.

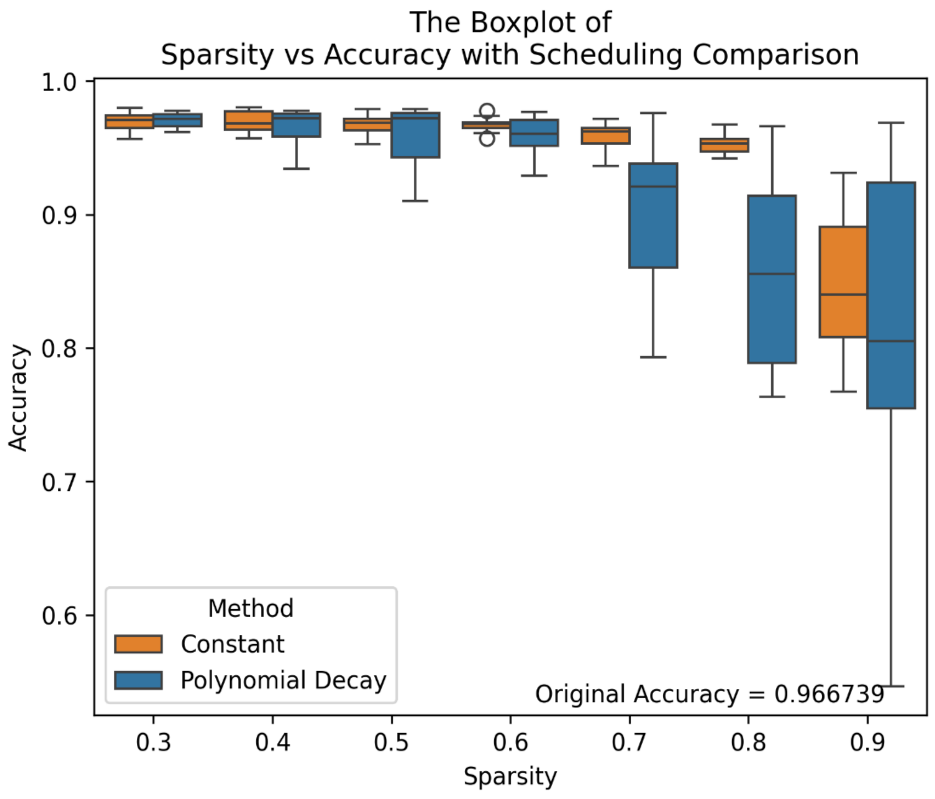

Figure 5. Comparison of the Influence of Two Scheduling Mechanisms in CNN Pruning on Accuracy with Varying Sparsity.

In the same test, we now observe the size of the compression model using the two scheduling methods. We also compared the model's size with the original model's size, which was 18.475 kB. There are no distribution of model size values for each ten iteration of tests because the model size in a deep learning training with constant input and hyperparameter is deterministic. Figure 6 shows a comparison of the bar charts. In contrast to accuracy, the size of the two scheduling models experiences a linear decrease in sparsity. Polynomial decay scheduling has a smaller model size at each sparsity than constant scheduling. The smallest size of this entire test results from polynomial decay scheduling with a sparsity of 0.9, namely 4.709 kB.

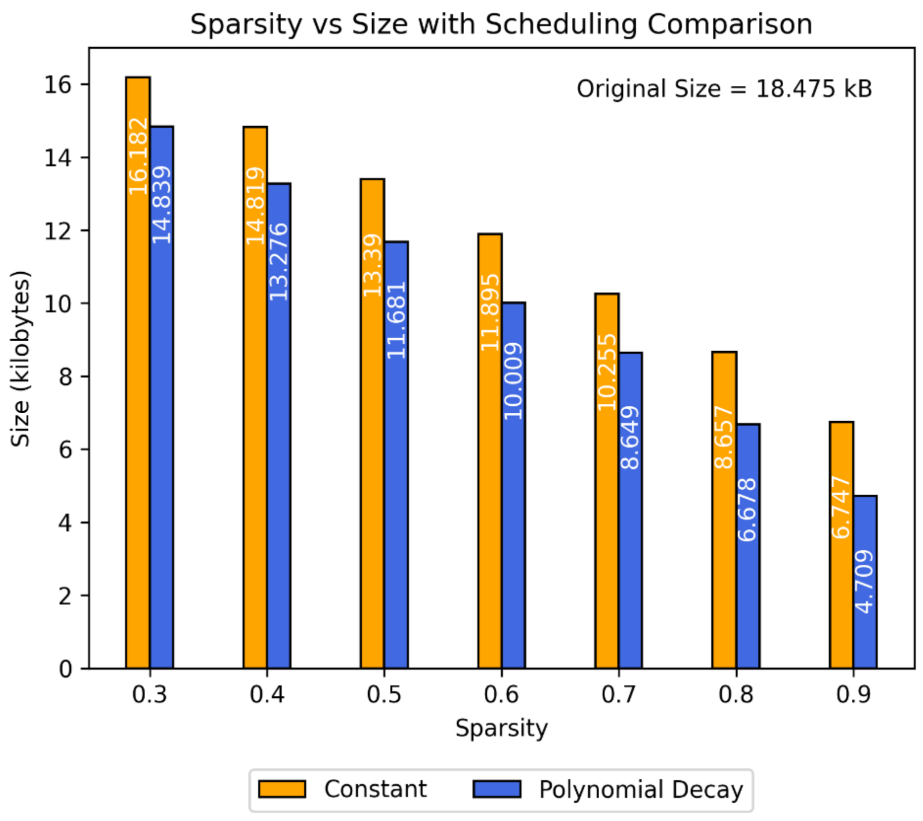

Figure 6. Comparison of the Influence of Two Scheduling Mechanisms in CNN Pruning on Model Size with Varying Sparsity.

Several aspects must be selected in selecting the optimum pruning result model, including reducing the accuracy and effectiveness of CR. For example, the compressed model resulting from polynomial decay scheduling with sparsity 0.9 gives the largest CR but gives too large an accuracy loss. We use a novel method called NG-Mean to objectively determine the optimal model. Based on this method, the optimal model is the model that uses constant scheduling with a sparsity value of 0.8. In this model, the mean accuracy is 0.960, and the size is 6.814 kB, which means there is a decrease in mean accuracy of 1.05% and CR of 2.71×.

2. Discussion

Several studies have used the corn leaf image dataset from Kaggle, which Smaranjit Ghose uploaded. One paper (Pamungkas et al., 2023) compared EffecientNet-B0 and ResNet-50 pre-trained models on this dataset and found that the optimum pre-trained model was EffecientNet-B0. However, the paper does not address the overfitting problem they encountered during training, as shown in the paper's learning curve. Another paper (Widianto et al., 2023) added feature extraction before using the CNN model in their training on the same dataset. These additional methods involve hue saturation value (HSV) coding, grayscale, region of interest, and grey level co-occurrence matrix (GLCM). These two papers have not discussed the architecture of corn disease detection.

Several papers have discussed architecture in corn disease detection. One paper (Shrestha et al., 2020) created an architecture for plant diseases, but it still uses a cloud architecture that is prone to delays in real-time systems. Another paper (Gajjar et al., 2022) discussed plant leaf disease detection using CNN and SSD MobileNet, where the model is drilled on a personal computer (PC), but the model is embedded on an edge computing device, namely the Nvidia Jetson. However, the paper does not apply model compression, which is vulnerable because the model complexity can be too high for such edge computing devices. Our research contribution is an architecture for deep learning-based corn disease detection that applies edge computing as well as pruning for model compression.

Several papers have used class weights in imbalanced datasets in image classification. One paper (Fernando & Tsokos, 2022) uses class weights to detect cyber-attacks where the imbalance condition is severe. The test results show that applying class weights provides a superior f1-score for detecting seven attacks. Another paper (Liu et al., 2023) embedded class weights in their new federated learning method for semi-supervised deep learning on medical images. The application of class weights in their method makes the performance of their method superior to five state-of-the-art methods. Our research uses class weights for the imbalance dataset in the corn leaf disease image dataset. The application of class weights makes the specificity and g-mean of the model better than without using class weights. No previous research utilized class weights that address the imbalance problem in corn leaf datasets. Our research contribution is a corn disease detection that uses class weights with optimum specificity and g-mean.

Several papers have applied grid search for hyperparameter optimization in deep learning models. A paper (Thanh, 2021) uses grid search to optimize a 1D-CNN model in a load forecasting case study. There were seven hyperparameters tuned in their research, where some hyperparameters had two values that were compared with 384 possible outputs. Another paper (Sharma & Vardhan, 2023) used grid search in a hybrid model between pre-trained VGG-Net and CNN which uses PCA to summarize the spectral features of diseased leaves. Through grid search, we found that the optimal CNN model for corn leaf disease detection uses hyperparameters, namely Adam optimization with LR of 0.001, batch size of 24, and epoch of 50. Our research contribution is the grid search optimization method for corn leaf disease detection using CNN.

Previous research (Nagarajan et al., 2019) have stated that the size of a deep learning model is deterministic when the input and hyperparameters are constant, which explains why the model size in this research remains constant for ten iterations. Then, several papers used various methods to find the optimal compressed model between decreasing accuracy and CR. A paper (Dong et al., 2023) uses a genetic algorithm (GA) to obtain an optimal model. The results of this research show that the GA method can minimize the decrease in accuracy due to pruning. Another paper (Jiang et al., 2023) which also uses pruning, uses knowledge distillation to move knowledge from an unpruned model to a pruned model. This method can increase CR while minimizing the decrease in accuracy. In this paper, we use NG-Mean, a novel method for minimizing accuracy loss, where with this method, we found that the optimal compressed model uses constant decay with a sparsity of 0.9 where the accuracy loss is 1.94% and the CR is 2.74×. Our research contribution is a novel method called NG-Mean for accuracy loss optimization in deep learning pruning. Table 3 summarizes our discussion while highlighting the contributions that have been made from this research.

Table 3. State-of-the-Art Research Comparison on CNN Pruning and Our Research Contribution.

Paper Reference	Plant Disease Detection	Smaranjit Ghose Dataset	System Architecture	Edge Computing	Class Weights	Grid Search	Accuracy Loss Optimization
(Pamungkas et al., 2023)	✓	✓	✗	✗	✗	✗	✗
(Widianto et al., 2023)	✓	✓	✗	✗	✗	✗	✗
(Shrestha et al., 2020)	✓	✗	✓	✗	✗	✗	✗
(Gajjar et al., 2022)	✓	✗	✓	✓	✗	✗	✗
(Fernando & Tsokos, 2022)	✗	✗	✗	✗	✓	✗	✗
(Liu et al., 2023)	✗	✗	✗	✗	✓	✗	✗
(Thanh, 2021)	✗	✗	✗	✗	✗	✓	✗
(Sharma & Vardhan, 2023)	✓	✗	✗	✗	✗	✓	✗
(Dong et al., 2023)	✗	✗	✗	✗	✗	✗	✓
(Jiang et al., 2023)	✗	✗	✗	✗	✗	✗	✓
Proposed Method	✓	✓	✓	✓	✓	✓	✓

While our study demonstrates promising results in using pruned CNN models for corn leaf disease detection using an edge computing approach, several limitations must be acknowledged. Firstly, the dataset from Kaggle is limited to images of specific diseases in corn, which may not capture full real-world conditions. This constraint could introduce biases. Additionally, the generalizability of our findings may be restricted due to the specific characteristics of the dataset and the controlled environment in which the data was collected. Future research should incorporate more diverse and larger datasets, possibly collected in varying environmental conditions and from different regions, to enhance the robustness and applicability of the models. Furthermore, investigating the application of our novel NG-Mean method and pruned CNN approach to other crops and diseases could provide valuable insights and broaden the impact of our work. Practical applications could also explore real-time deployment scenarios, assessing the system's performance in field conditions and its integration with existing agricultural technologies to provide comprehensive support to farmers.

CONCLUSIONS AND RECOMMENDATIONS

In this research, we developed an architecture for corn disease detection that uses edge computing. This architecture uses a smartphone to capture images of corn leaves. We use a CNN architecture with a pruning method for edge-based disease detection. Several methods, such as class weights, grid search, and a novel method called NG-Mean, are used to optimize the deep learning model. The test results show that class weights can optimize specificity and g-mean on the imbalanced dataset, with values of 0.995 and 0.983, respectively. Then, the grid search results can optimize the optimization method hyperparameters, learning rate, batch size, and epoch to get the best accuracy of 0.947. The results of applying pruning produce several models with variations in sparsity and scheduling method. As a result of the novel NG-mean method, we succeeded in finding the optimum compressed model with constant scheduling and sparsity of 0.8 with a mean accuracy loss of 1.05% and a CR of 2.71×.

For future recommendations, we suggest implementing the edge computing architecture that we proposed. We also propose implementing the pruned CNN model in this research to run on smartphones using applications such as TensorFlow Light (TF Light). Finally, we propose the use of other model compression methods for deep learning, such as quantization, knowledge distillation, and low-rank factorization, to obtain more optimal accuracy loss.

ACKNOWLEDGEMENT

We thank Smaranjit Ghose for uploading the corn disease image dataset to Kaggle containing leaves with blight, common rust, grey leaf spots, and healthy conditions. With this dataset, we succeeded in finding contributions to CNN pruning. We also thank the Directorate of Research and Community Service (PPM) Telkom University for funding this research.

REFERENCES

Ahia, O., Kreutzer, J., & Hooker, S. (2021). *The Low-Resource Double Bind: An Empirical Study of Pruning for Low-Resource Machine Translation* (arXiv:2110.03036). arXiv. https://doi.org/10.48550/arXiv.2110.03036

Al-Adhaileh, M. H., Verma, A., Aldhyani, T. H., & Koundal, D. (2023). Potato blight detection using fine-tuned CNN architecture. *Mathematics*, *11*(6), 1516.

Anim-Ayeko, A. O., Schillaci, C., & Lipani, A. (2023). Automatic blight disease detection in potato (Solanum tuberosum L.) and tomato (Solanum lycopersicum, L. 1753) plants using deep learning. *Smart Agricultural Technology*, *4*, 100178. https://doi.org/10.1016/j.atech.2023.100178

Ashwini, C., & Sellam, V. (2023). Analyzing The Prediction Accuracy Of Corn Leaf Diseases Using A Pre-Trained Network Model. *2023 International Conference on Advancement in Computation & Computer Technologies (InCACCT)*, 87–91. https://doi.org/10.1109/InCACCT57535.2023.10141840

Choudhary, K., Nersisyan, N., Lin, E., Chandrasekaran, S., Mayani, R., Pottier, L., Murillo, A. P., Virdone, N. K., Kee, K., & Deelman, E. (2022). Application of Edge-to-Cloud Methods Toward Deep Learning. *2022 IEEE 18th International Conference on E-Science (e-Science)*, 415–416. https://doi.org/10.1109/eScience55777.2022.00065

Cui, L., Su, X., Ming, Z., Chen, Z., Yang, S., Zhou, Y., & Xiao, W. (2022). CREAT: Blockchain-Assisted Compression Algorithm of Federated Learning for Content Caching in Edge Computing. *IEEE Internet of Things Journal*, *9*(16), 14151–14161. https://doi.org/10.1109/JIOT.2020.3014370

Dagwale, S. S., & Adakane, P. (2023). Prediction of Leaf Species & Disease Using Ai for Various Plants. *Int. J. Multidiscip. Res*, *5*, 23034169.

Degani, O., Movshowitz, D., Dor, S., Meerson, A., & Rabinovitz, O. (2018). Evaluating Azoxystrobin Seed Coating Against Maize Late Wilt Disease Using a Sensitive qPCR-Based Method. *Plant Disease*, *103*. https://doi.org/10.1094/PDIS-05-18-0759-RE

Dong, X., Li, B., & Song, Y. (2023). *An Optimization Method For Pruning Rates of Each Layer in CNN Based on the GA-SMSM*. https://doi.org/10.21203/rs.3.rs-2738666/v1

Escobar, L., Gallardo, P., González-Anaya, J., González, J. L., Montúfar, G., & Morales, A. H. (2022). *Enumeration of max-pooling responses with generalized permutohedra*. https://doi.org/10.48550/ARXIV.2209.14978

Fan, W., Chu, F., Wang, H., & Yu, P. S. (2002). Pruning and dynamic scheduling of cost-sensitive ensembles. *AAAI/IAAI*, 146–151. https://cdn.aaai.org/AAAI/2002/AAAI02-023.pdf

Fernando, K. R. M., & Tsokos, C. P. (2022). Dynamically Weighted Balanced Loss: Class Imbalanced Learning and Confidence Calibration of Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, *33*(7), 2940–2951. https://doi.org/10.1109/TNNLS.2020.3047335

Gajjar, R., Gajjar, N., Thakor, V. J., Patel, N. P., & Ruparelia, S. (2022). Real-time detection and identification of plant leaf diseases using convolutional neural networks on an embedded platform. *The Visual Computer*, *38*(8), 2923–2938. https://doi.org/10.1007/s00371-021-02164-9

Gangopadhyay, B., Dasgupta, P., & Dey, S. (2023). Safety Aware Neural Pruning for Deep Reinforcement Learning (Student Abstract). *Proceedings of the AAAI Conference on Artificial Intelligence*, *37*(13), 16212–16213. https://doi.org/10.1609/aaai.v37i13.26966

Jia, Y., Liu, B., Dou, W., Xu, X., Zhou, X., Qi, L., & Yan, Z. (2022). CroApp: A CNN-based resource optimization approach in edge computing environment. *IEEE Transactions on Industrial Informatics*, *18*(9), 6300–6307.

Jiang, H., Zhang, L. L., Li, Y., Wu, Y., Cao, S., Cao, T., Yang, Y., Li, J., Yang, M., & Qiu, L. (2023). *Accurate and Structured Pruning for Efficient Automatic Speech Recognition*. https://doi.org/10.48550/ARXIV.2305.19549

Joardar, B. K., Doppa, J. R., Li, H., Chakrabarty, K., & Pande, P. P. (2023). ReaLPrune: ReRAM Crossbar-Aware Lottery Ticket Pruning for CNNs. *IEEE Transactions on Emerging Topics in Computing*, *11*(2), 303–317. https://doi.org/10.1109/TETC.2022.3223630

Kurtic, E., Campos, D., Nguyen, T., Frantar, E., Kurtz, M., Fineran, B., Goin, M., & Alistarh, D. (2022). *The Optimal BERT Surgeon: Scalable and Accurate Second-Order Pruning for Large Language Models*. https://doi.org/10.48550/ARXIV.2203.07259

Lei, F., Liu, X., Dai, Q., & Ling, B. W.-K. (2020). Shallow convolutional neural network for image classification. *SN Applied Sciences*, *2*(1), 97. https://doi.org/10.1007/s42452-019-1903-4

Liu, W., Mo, J., & Zhong, F. (2023). Class Imbalanced Medical Image Classification Based on Semi-Supervised Federated Learning. *Applied Sciences*, *13*(4), Article 4. https://doi.org/10.3390/app13042109

Luo, J.-H., Zhang, H., Zhou, H.-Y., Xie, C.-W., Wu, J., & Lin, W. (2018). ThiNet: Pruning CNN filters for a thinner net. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*(10), 2525–2538.

Moon, S., Byun, Y., Park, J., Lee, S., & Lee, Y. (2019). Memory-reduced network stacking for edge-level CNN architecture with structured weight pruning. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *9*(4), 735–746.

Nagarajan, P., Warnell, G., & Stone, P. (2019). *Deterministic Implementations for Reproducibility in Deep Reinforcement Learning* (arXiv:1809.05676). arXiv. http://arxiv.org/abs/1809.05676

Pamungkas, W. G., Wardhana, M. I. P., Sari, Z., & Azhar, Y. (2023). Leaf Image Identification: CNN with EfficientNet-B0 and ResNet-50 Used to Classified Corn Disease. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *7*(2), 326–333.

Pandey, G., & Dukkipati, A. (2017). Unsupervised feature learning with discriminative encoder. *2017 IEEE International Conference on Data Mining (ICDM)*, 367–376. https://ieeexplore.ieee.org/abstract/document/8215509/

Pane, S. F., Ramdan, J., Putrada, A. G., Fauzan, M. N., Awangga, R. M., & Alamsyah, N. (2022). A Hybrid CNN-LSTM Model With Word-Emoji Embedding For Improving The Twitter Sentiment Analysis on Indonesia's PPKM Policy. *2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, 51–56. https://doi.org/10.1109/ICITISEE57756.2022.10057720

Prabowo, S., Putrada, A. G., Oktaviani, I. D., & Abdurohman, M. (2023). Camera-Based Smart Lighting System that complies with Indonesia's Personal Data Protection Act. *2023 International Conference on Advancement in Data Science, E-Learning and Information System (ICADEIS)*, 1–6. https://ieeexplore.ieee.org/abstract/document/10271086/

Putrada, A. G., Abdurohman, M., Perdana, D., & Nuha, H. H. (2023a). EdgeSL: Edge-Computing Architecture on Smart Lighting Control With Distilled KNN for Optimum Processing Time. *IEEE Access*, *11*, 64697–64712. https://doi.org/10.1109/ACCESS.2023.3288425

Putrada, A. G., Abdurohman, M., Perdana, D., & Nuha, H. H. (2023b). Shuffle Split-Edited Nearest Neighbor: A Novel Intelligent Control Model Compression for Smart Lighting in Edge Computing Environment. In *Information Systems for Intelligent Systems: Proceedings of ISBM 2022* (pp. 219–227). Springer.

Putrada, A. G., Abdurohman, M., Perdana, D., & Nuha, H. H. (2024). Q8KNN: A Novel 8-Bit KNN Quantization Method for Edge Computing in Smart Lighting Systems with NodeMCU. In K. Arai (Ed.), *Intelligent Systems and Applications* (Vol. 824, pp. 598–615). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-47715-7_41

Putrada, A. G., Alamsyah, N., Pane, S. F., Fauzan, M. N., & Perdana, D. (2023). Knowledge Distillation for a Lightweight Deep Learning-Based Indoor Positioning System on Edge Environments. *2023 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, 370–375. https://ieeexplore.ieee.org/abstract/document/10220996/?casa_token=f_dJRWL4VpkAAAAA:t-H9-HfnggTsdqhYXNcUObfQsD7d-2MnLTht-eaCMGNKrMt1r3eH5lrj5pkVmdgh9WUMTOGTkpk

Qin, L., & Sun, J. (2023). Model Compression for Data Compression: Neural Network Based Lossless Compressor Made Practical. *2023 Data Compression Conference (DCC)*, 52–61. https://doi.org/10.1109/DCC55655.2023.00013

Sharma, S., & Vardhan, M. (2023). Hyperparameter Tuned Hybrid Convolutional Neural Network (H-CNN) for Accurate Plant Disease Classification. *2023 International Conference on Communication, Circuits, and Systems (IC3S)*, 1–6. https://doi.org/10.1109/IC3S57698.2023.10169257

Shrestha, U., Peter, D. T. J., Manandhar, S., Rajbanshi, S., Padal, S., & Student, U. (2020). *System Design for Identification of Plant Leaf Diseases using Deep Learning for Web and Mobile Platform*. https://www.semanticscholar.org/paper/System-Design-for-Identification-of-Plant-Leaf-Deep-Shrestha-Peter/97948b361657578fe0619fa33cb36e4d2efad442

Siahroudi, S. K., & Kudenko, D. (2023). An effective single-model learning for multi-label data. *Expert Systems with Applications*, *232*, 120887. https://doi.org/10.1016/j.eswa.2023.120887

Tang, J., Liu, S., Liu, L., Yu, B., & Shi, W. (2020). LoPECS: A Low-Power Edge Computing System for Real-Time Autonomous Driving Services. *IEEE Access*, *8*, 30467–30479. https://doi.org/10.1109/ACCESS.2020.2970728

Thanh, T. (2021). Grid Search of Convolutional Neural Network model in the case of load forecasting. *Archives of Electrical Engineering*, *70*. https://doi.org/10.24425/aee.2021.136050

Velmurugan, S., Moorthy, R. S., & Angel, S. (2023). Computer Vision based Plant Disease Detection using Machine Learning Technique. *International Journal*, *11*(7). https://www.academia.edu/download/104134478/ijeter021172023.pdf

Vocaturo, E., Zumpano, E., & Veltri, P. (2018). Image pre-processing in computer vision systems for melanoma detection. *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 2117–2124. https://doi.org/10.1109/BIBM.2018.8621507

Widianto, B., Utami, E., & Ariatmanto, D. (2023). Identifikasi Penyakit Tanaman Jagung Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *Techno. Com*, *22*(3), 599–608.

Ye, Y., Li, S., Liu, F., Tang, Y., & Hu, W. (2020). EdgeFed: Optimized federated learning based on edge computing. *IEEE Access*, *8*, 209191–209198.

Yucky, E. D. D., Putrada, A. G., & Abdurohman, M. (2021). IoT drone camera for a paddy crop health detector with RGB comparison. *2021 9th International Conference on Information and Communication Technology (ICoICT)*, 155–159. https://ieeexplore.ieee.org/abstract/document/9527421/

Zhan, T. (2022). Hyper-Parameter Tuning in Deep Neural Network Learning. *Artificial Intelligence and Applications*, 95–101. https://doi.org/10.5121/csit.2022.121809