

CornLeafNet: Disease-Area-Based Corn Leaf Disease Classification Using Convolutional Neural Networks

Herfandi Herfandi^{1*}, Eri Sasmita Susanto², Fahri Hamdani³, Jonathan Afriliansyah⁴

Informatics Study Program, Faculty of Systems Engineering, Universitas Teknologi Sumbawa, Sumbawa, Indonesia

¹herfandi@uts.ac.id, ²eri.sasmita.susanto@uts.ac.id, ³fahri.hamdani@uts.ac.id, ⁴jonafriliansyah3@gmail.com

ARTICLE INFO

Article history:

Received : 2026-07-01

Revised : 2026-07-10

Accepted : 2026-07-21

Kata Kunci:

CornLeafNet; penyakit daun jagung; klasifikasi berbasis area gejala; Convolutional Neural Network (CNN); bounding box; ONNX.

Keywords:

CornLeafNet; corn leaf disease; disease-area-based classification; Convolutional Neural Network (CNN); bounding box; ONNX.

ABSTRAK

Penyakit daun jagung dapat menurunkan produktivitas tanaman karena mengganggu proses fotosintesis dan pertumbuhan. Identifikasi manual pada lahan berskala luas masih memiliki keterbatasan karena bergantung pada keahlian pengamat, kemiripan gejala antarpenyakit, dan variasi kondisi lapangan. Penelitian ini mengusulkan CornLeafNet, yaitu model Custom Convolutional Neural Network untuk klasifikasi penyakit daun jagung berbasis area gejala. Dataset yang digunakan berupa citra daun jagung beranotasi XML, kemudian area penyakit diekstraksi melalui parsing anotasi dan cropping bounding box agar model berfokus pada bagian daun yang menunjukkan gejala. CornLeafNet dikembangkan untuk mengklasifikasikan tiga penyakit, yaitu Grey Leaf Spot, Corn Rust, dan Leaf Blight. Hasil evaluasi menunjukkan bahwa model mencapai validation accuracy 97,66% dan testing accuracy 98,60%, dengan precision, recall, dan F1-score masing-masing sebesar 0,9860. Model terbaik juga dikonversi ke format ONNX dan diterapkan pada prototipe web untuk klasifikasi berbasis gambar dan video. Hasil pengujian menunjukkan bahwa fungsi utama sistem berjalan sesuai rancangan, sehingga CornLeafNet berpotensi menjadi model pendukung identifikasi awal penyakit daun jagung secara otomatis dan praktis.

ABSTRACT

Corn leaf diseases can reduce crop productivity by disrupting photosynthesis and plant growth. Manual identification in large-scale fields remains limited due to its dependence on observer expertise, visual similarity among disease symptoms, and variations in field conditions. This study proposes CornLeafNet, a Custom Convolutional Neural Network model for disease-area-based corn leaf disease classification. The dataset consists of XML-annotated corn leaf images, from which disease-affected regions were extracted through annotation parsing and bounding box-based cropping to focus the model on symptomatic leaf areas. CornLeafNet was developed to classify three disease categories: Grey Leaf Spot, Corn Rust, and Leaf Blight. The model achieved a validation accuracy of 97.66% and a testing accuracy of 98.60%, with precision, recall, and F1-score values of 0.9860, respectively. The best-performing model was converted into ONNX format and deployed in a web-based prototype for image- and video-based classification. The testing results showed that all core system functions operated as expected, indicating that CornLeafNet has potential as an automatic and practical support model for early corn leaf disease identification.



This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.

Copyright © 2025 by Author. Published by GWS Tech Solution.

*Corresponding author.

E-mail addresses: herfandi@uts.ac.id

1. INTRODUCTION

Corn is a strategic agricultural commodity that plays an important role in supporting food security, the animal feed industry, and community economic activities [1], [2]. Corn productivity is strongly influenced by plant health, particularly leaf condition. Since leaves are essential for photosynthesis, foliar diseases may directly affect plant growth and reduce crop yield [3], [4].

Several major corn leaf diseases, including Grey Leaf Spot, Corn Rust, and Leaf Blight, are known to cause productivity losses when they are not detected and managed at an early stage [4], [5], [6]. These diseases often present visually similar symptoms, such as spots, lesions, discoloration, and leaf tissue damage. Such visual similarities make manual identification challenging, especially when monitoring is conducted across large agricultural areas. Therefore, an identification approach that can detect corn leaf diseases quickly, accurately, and consistently is required.

The development of deep learning [7], particularly Convolutional Neural Networks (CNNs) [8], has provided substantial opportunities for plant image processing and classification. CNNs are capable of automatically learning visual features, including texture, color, shape, and damage patterns on leaves. Compared with conventional approaches that require manually designed feature extraction, CNNs can directly extract relevant features from input images. This capability makes CNN a suitable approach for supporting automatic plant disease classification, including corn leaf disease identification.

The need for an automatic identification model is also relevant to the agricultural context of Sumbawa Regency, particularly Sumbawa Besar, where corn is cultivated on a large scale. The Central Statistics Agency of West Nusa Tenggara Province reported that in 2024, the harvested corn area in NTB reached 173.76 thousand hectares, with total production of 1.21 million tons. At the Sumbawa Regency level, data as of March 2025 showed a corn planting target of 105,169 hectares, a realized planting area of 79,105 hectares, and production reaching 574,302 tons [9], [10]. This production scale indicates that manual plant health monitoring is less efficient because it requires considerable time, labor, and adequate visual expertise.

Previous studies have demonstrated the effectiveness of CNNs in image-based plant disease classification. Ulfa Khaira et al. (2024) developed an Android-based corn disease detection application using CNN-MobileNetV2 and achieved an accuracy of 99–100% [11]. However, the study did not utilize a disease-area-based dataset. Muhammad Naufal Anugrah and Nur Rachmat (2025) applied MobileNetV4 transfer learning to 4,000 corn leaf images and obtained a validation accuracy of 95.25% and an F1-score of 92.00% [12]. Nevertheless, their study did not emphasize bounding box-based symptom

extraction. Muhammad Rijal, Andi Muhammad Yani, and Abdul Rahman (2024) applied CNN for rice leaf disease classification [13], but their research object was not corn leaves. Kusdatul Komariyah, Ratri Enggar Pawening, and Moh Furqan (2025) developed a web-based corn stalk disease classification system using CNN-MobileNet [14], yet the study focused on corn stalks and used a limited dataset. Meanwhile, Imam Wirabowo and Indah Susilawati (2025) developed a sequential CNN for corn leaf disease detection [15], but the approach still relied on general leaf images without disease-area extraction based on annotation.

Based on the review of previous studies, two main research gaps can be identified. First, most existing studies still use whole-leaf images, which may cause the model to learn irrelevant background features that are not directly associated with disease symptoms. Second, several studies have not specifically emphasized the use of annotated disease regions as the main input for classification. As a result, the model may not optimally focus on the most informative regions of the leaf image. Therefore, an approach is required that directs model learning toward disease-affected areas while maintaining high classification performance.

Accordingly, this study proposes CornLeafNet, a Custom Convolutional Neural Network model for disease-area-based corn leaf disease classification. The main contributions of this study are threefold. First, this study constructs a disease-area-based classification dataset through XML annotation parsing and bounding box-based cropping so that the model can focus on leaf regions showing disease symptoms. Second, this study develops the CornLeafNet CNN architecture to classify three corn leaf diseases, namely Grey Leaf Spot, Corn Rust, and Leaf Blight. Third, this study evaluates the best-performing model and deploys it in a web-based prototype to demonstrate its practical applicability for image-based and video-based disease classification.

2. METHOD

This study was conducted through systematic stages to develop a disease-area-based corn leaf disease classification model using Convolutional Neural Networks. The proposed model, named CornLeafNet, was designed to recognize three disease classes: Grey Leaf Spot, Corn Rust, and Leaf Blight.

In general, the research stages consisted of literature review, dataset collection and preparation [16], construction of a disease-area-based classification dataset, model architecture design, model training and evaluation, model conversion into ONNX format, and deployment of the trained model in a web-based prototype for practical testing.

2.1. CornLeafNet Training Workflow

The development of CornLeafNet began with a corn leaf image dataset annotated in XML format. These

annotations were used to obtain the location of disease-affected regions through bounding box coordinates. The XML files were parsed to extract bounding box information, after which the regions containing disease symptoms were cropped from the original images.

Bounding box-based cropping was applied to prevent the model from learning the entire leaf image indiscriminately. Instead, this approach allowed the model to focus on regions that contain visible disease symptoms. After cropping, the extracted images were resized to 224×224 pixels and normalized according to the model input requirements.

The cropped dataset was then divided into training, validation, and testing subsets. Image augmentation was applied to the training subset to increase data variability and improve the model's generalization ability under different visual conditions. The augmentation techniques included rotation, width shift, height shift, shear transformation, zoom, horizontal flip, vertical flip, brightness adjustment, and channel shift.

Model training was performed using the Adam optimizer with an initial learning rate of 0.001, a batch size of 32, and a maximum of 50 epochs. To maintain training stability and obtain the best model weights, ModelCheckpoint, EarlyStopping, and ReduceLROnPlateau mechanisms were applied. After training, the model was evaluated using accuracy, precision, recall, F1-score, and confusion matrix. The best-performing model was then converted into ONNX format to support deployment and inference testing.

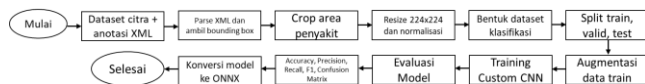


Fig. 1. CornLeafNet training workflow

2.2. CornLeafNet Architecture

CornLeafNet was developed using a hierarchical Convolutional Neural Network architecture to classify corn leaf diseases based on visual patterns in disease-affected regions. The model receives cropped disease-area images with an input size of $224 \times 224 \times 3$. These images are processed through five feature extraction blocks and one classifier head. The implemented architecture contains 9 convolutional layers, 7 batch normalization layers, 4 max pooling layers, 1 global average pooling layer, 6 dropout layers, 2 hidden dense layers, and 1 softmax output layer. The model has 2,754,211 total parameters, consisting of 2,750,691 trainable parameters and 3,520 non-trainable parameters. The detailed architecture of CornLeafNet is presented in Table 1.

Table 1. Detailed CornLeafNet Architecture

Stage	Layer Type	Main Parameters	Activation	Regularization or Operation
Input	Input layer	$224 \times 224 \times 3$	None	Cropped disease-area image
Block 1	Conv2D $\times 2$	32 filters, 3×3 , same padding	ReLU	BatchNorm, MaxPooling 2×2 , Dropout 0.25
Block 2	Conv2D $\times 2$	64 filters, 3×3 , same padding	ReLU	BatchNorm, MaxPooling 2×2 , Dropout 0.25
Block 3	Conv2D $\times 2$	128 filters, 3×3 , same padding	ReLU	BatchNorm, MaxPooling 2×2 , Dropout 0.25
Block 4	Conv2D $\times 2$	256 filters, 3×3 , same padding	ReLU	BatchNorm, MaxPooling 2×2 , Dropout 0.25
Block 5	Conv2D	512 filters, 3×3 , same padding	ReLU	BatchNorm, GlobalAveragePooling
Classifier 1	Dense	512 neurons	ReLU	BatchNorm, Dropout 0.50
Classifier 2	Dense	256 neurons	ReLU	BatchNorm, Dropout 0.30
Output	Dense	3 neurons	Softmax	Three-class prediction

As shown in Table 1, the first to fourth feature extraction blocks use two Conv2D layers in each block. The number of filters increases gradually from 32, 64, 128, to 256. Each convolutional layer uses a 3×3 kernel, ReLU activation, and same padding. This structure allows the model to learn visual features progressively, starting from low-level texture and color patterns to more complex lesion representations. Each block also uses batch normalization, max pooling, and dropout to stabilize training, reduce spatial dimensions, and minimize overfitting.

The fifth feature extraction block uses one Conv2D layer with 512 filters, followed by batch normalization and global average pooling. Global average pooling converts the spatial feature maps into a compact feature vector before classification. This approach reduces the number of parameters compared with a flatten-based classifier. The classifier head consists of two hidden dense layers with 512 and 256 neurons. Both dense layers use ReLU activation and are followed by batch normalization and dropout. The output layer consists of three neurons with softmax activation, representing Grey Leaf Spot, Corn Rust, and Leaf Blight. The final class is selected based on the highest probability score. The model training configuration is presented in Table 2.

Table 2. CornLeafNet Training Parameters

Parameter	Value
Optimizer	Adam
Initial learning rate	0.001
Loss function	Sparse categorical cross-entropy
Batch size	32
Maximum epochs	50
Training samples	6643
Validation samples	1424
Testing samples	1424
ModelCheckpoint monitor	Validation accuracy
EarlyStopping monitor	Validation loss
EarlyStopping patience	10
ReduceLROnPlateau monitor	Validation accuracy
ReduceLROnPlateau factor	0.2
Minimum learning rate	1×10^{-7}

As presented in Table 2, CornLeafNet was trained using the Adam optimizer with an initial learning rate of 0.001. The sparse categorical cross-entropy loss function was used because the class labels were encoded as integer indices in a multi-class classification task. The model was trained with a batch size of 32 and a maximum of 50 epochs. To improve training stability, ModelCheckpoint was used to save the best model weights based on validation accuracy. EarlyStopping monitored validation loss with a patience value of 10 and restored the best weights when the model stopped improving. ReduceLROnPlateau monitored validation accuracy and reduced the learning rate by a factor of 0.2 when the validation performance reached a plateau, with a minimum learning rate of 1×10^{-7} .

Image augmentation was applied to the training dataset to improve model generalization under different visual conditions. The augmentation process introduced variations in position, orientation, scale, brightness, and color distribution. These variations are important because corn leaf images may be captured under different lighting conditions and acquisition angles in real field environments. The image augmentation parameters are presented in Table 3.

Table 3. Image Augmentation Parameters

Augmentation Technique	Parameter Value
Rotation range	30 degrees
Width shift range	0.3
Height shift range	0.3
Shear range	0.2
Zoom range	0.3
Horizontal flip	TRUE
Vertical flip	TRUE
Brightness range	0.7 to 1.3
Channel shift range	0.2
Fill mode	Reflect

Rescale

1/255

As shown in Table 3, the augmentation process included rotation, width shift, height shift, shear transformation, zoom, horizontal flip, vertical flip, brightness adjustment, and channel shift. Brightness adjustment helped the model learn disease patterns under different lighting conditions, while rotation, flipping, shifting, and zooming supported the model in handling variations in image acquisition angles and lesion positions. The pixel values were also rescaled by 1/255 to normalize the image input before training. This augmentation strategy helped CornLeafNet learn more robust visual features from disease-affected regions and reduced the risk of overfitting to a specific image condition.

Figure 2 illustrates the overall architecture of CornLeafNet. As shown in Figure 2, the model receives cropped disease-area images with a size of $224 \times 224 \times 3$ as input. The input is then processed sequentially through five feature extraction blocks. The first to fourth blocks use convolutional layers, batch normalization, max pooling, and dropout, while the fifth block uses convolutional processing followed by global average pooling. The extracted features are then passed into two classifier heads with 512 and 256 neurons before being classified by the softmax output layer into three disease classes. This architecture shows that CornLeafNet was designed to extract visual features progressively from disease-affected regions and convert them into class probabilities for Grey Leaf Spot, Corn Rust, and Leaf Blight.

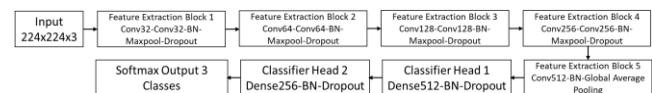


Fig. 2. CornLeafNet architecture

2.3. Model Deployment Design

Although the primary focus of this study is the development and evaluation of CornLeafNet as a CNN-based classification model, the trained model was also deployed in a web-based prototype to assess its practical applicability. The deployment process used ONNX Runtime as the inference engine. The system supports two types of input: image and video.

In the image-based scenario, users upload a corn leaf image through the prototype interface. The system then loads the ONNX model, performs preprocessing by resizing the image to 224×224 pixels, normalizes pixel values to the range of 0–1, and executes the inference process. The inference output consists of probability values for each class. The class with the highest probability is selected as the final prediction and displayed along with the confidence value and disease information.

In the video-based scenario, the system extracts

frames from the uploaded video at an interval of approximately one frame per second. Each extracted frame is processed using the same preprocessing steps as image input and then classified using the CornLeafNet model. The prediction result for each frame is displayed as a disease class with its corresponding confidence value. After all frames are analyzed, the system summarizes the prediction results to determine the dominant class as the overall summary. This design enables the model to be tested not only on single images but also on video input through frame-based inference.

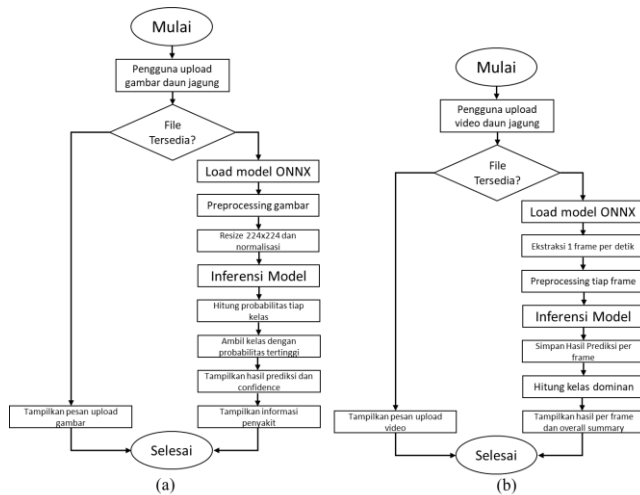


Fig. 3. CornLeafNet deployment workflow for (a) image input and (b) video input

2.4. Implementation Environment

The system was developed using the Python programming language. ONNX Runtime was used as the inference engine, while OpenCV, PIL, and NumPy were used for image processing and numerical computation. Implementation and testing were conducted on a Dell Precision 3660 workstation running Windows 11 Pro 64-bit, equipped with an Intel Core i7-12700 processor, 16 GB RAM, and an NVIDIA RTX A2000 GPU with 6 GB VRAM. These specifications were considered sufficient to support stable and efficient classification processes [18].

Table 4. Implementation Environment Specifications

Component	Specification
Operating system	Windows 11 Pro 64-bit
Device	Dell Precision 3660
Processor	Intel Core i7-12700
RAM	16 GB
GPU	NVIDIA RTX A2000, 6 GB VRAM
Programming language	Python
Inference engine	ONNX Runtime
Model format	ONNX

3. RESULT AND DISCUSSION

3.1. Dataset Characteristics

The dataset used in this study consisted of 9,411 XML-annotated corn leaf images. The dataset was organized into training, validation, and testing subsets, with a uniform image size of 416×416 pixels. The dataset was selected because it contained high-quality annotations, variations in image acquisition angles, and diverse lighting conditions. These variations are important because they allow the model to learn disease patterns under visual conditions that are more representative of real field environments.

After cropping based on disease-area bounding boxes, 9,491 classification image samples were obtained. These samples consisted of 3,015 Grey Leaf Spot images, 3,060 Corn Rust images, and 3,416 Leaf Blight images. The difference between the number of original images and cropped samples occurred because one image could contain more than one annotated disease region.

Each disease class has distinct visual characteristics. Grey Leaf Spot is generally characterized by elongated pale or grayish spots. Corn Rust is indicated by brown to reddish pustules on the leaf surface. Meanwhile, Leaf Blight is characterized by elongated necrotic lesions. These visual differences form the basis for the model to learn and distinguish among the three disease classes.

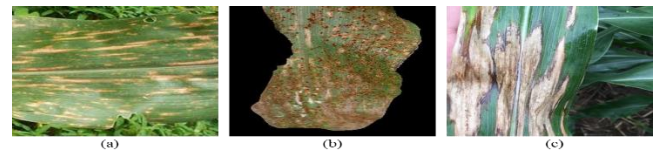


Fig. 4. Sample images of corn leaf diseases: (a) Grey Leaf Spot, (b) Corn Rust, and (c) Leaf Blight

3.2. Model Training and Evaluation Results

The cropped dataset was divided into 6,643 training samples, 1,424 validation samples, and 1,424 testing samples. The training process was stopped at the 36th epoch through the early stopping mechanism, while the best model weights were obtained at the 26th epoch. At the best epoch, the model achieved a validation accuracy of 97.66% and a validation loss of 0.0663.

These results indicate that the training strategy was able to maintain model performance while reducing the risk of overfitting. Evaluation on the testing dataset showed that CornLeafNet achieved an accuracy of 98.60%, a loss of 0.0450, and precision, recall, and F1-score values of 0.9860, respectively. The summary of the model evaluation results is presented in Table 5.

Table 5. Summary of CornLeafNet Model Evaluation Results

Loss	Accuracy	Precision	Recall	F1-score
0.045	98.60%	0.986	0.986	0.986

The evaluation results indicate that the hierarchical CNN architecture of CornLeafNet was able to effectively extract visual patterns associated with corn leaf diseases. The training and validation accuracy curves showed a stable increasing trend during training. Meanwhile, the training and validation loss curves decreased significantly in the early training phase and tended to stabilize in subsequent epochs. This pattern indicates that the optimization process was performed effectively.

The per-class evaluation also demonstrated consistent performance. Grey Leaf Spot achieved a precision of 0.98, recall of 0.99, and F1-score of 0.98. Corn Rust achieved a precision of 1.00, recall of 0.99, and F1-score of 1.00. Leaf Blight achieved a precision of 0.98, recall of 0.98, and F1-score of 0.98.

The confusion matrix results show that most testing samples were correctly classified. Misclassification occurred only in a small number of samples, particularly between Grey Leaf Spot and Leaf Blight. This may be attributed to the visual similarity between the two diseases, especially in terms of elongated spots or lesions on the leaves. In addition to symptom similarity, lighting conditions and image acquisition angles may also influence model performance because they affect the visibility of lesion color, texture, contrast, and symptom boundaries. Images captured under uneven lighting, low brightness, shadowed areas, or oblique angles may reduce the clarity of disease patterns and increase the chance of incorrect classification.

Despite these variations, CornLeafNet maintained high testing performance. This result indicates that the model was able to learn discriminative visual features from disease-affected regions. The use of bounding box-based cropping helped the model focus on symptomatic areas rather than irrelevant background regions. In addition, image augmentation during training, including rotation, brightness adjustment, channel shift, zoom, horizontal flip, and vertical flip, increased the variability of training samples and improved the model's generalization ability under different visual conditions. The per-class accuracy remained high, reaching approximately 98.7% for Grey Leaf Spot, 99.1% for Corn Rust, and 98.0% for Leaf Blight. These results indicate that CornLeafNet can classify corn leaf diseases effectively, although further evaluation using controlled lighting and angle-based testing scenarios is recommended for future work.

The best-performing model was subsequently converted into ONNX format using opset 13 and was successfully verified as a valid model. A comparison between the TensorFlow and ONNX model predictions showed very small differences, with maximum difference and mean difference values approaching 0.0000. This confirms that the converted ONNX model produced outputs equivalent to the original model and was therefore suitable for deployment.

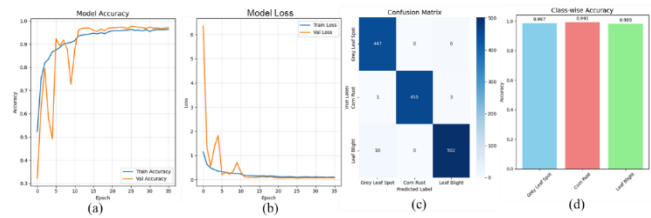


Fig. 5. Visualization of CornLeafNet training and evaluation results: (a) accuracy curve, (b) loss curve, (c) confusion matrix, and (d) per-class accuracy

3.3. Model Training and Evaluation Results

The interaction between users and the deployed prototype is represented through a use case diagram [19]. The diagram illustrates three main user activities: accessing the main page, uploading corn leaf files in image or video format, and viewing classification results. This design indicates that the prototype was developed with a simple interface that focuses directly on the disease identification process.

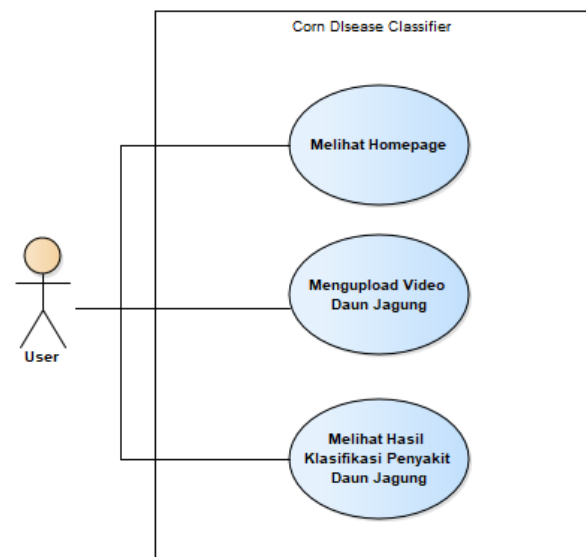


Fig. 6. Use case diagram system

3.4. Prototype Implementation Results

The trained CornLeafNet model, which had been converted into ONNX format, was deployed in a web-based prototype for corn leaf disease classification. At the initial stage, the system loads the ONNX model and creates an inference session. The system is designed to use GPU resources when available, while CPU is used as an alternative when GPU support is unavailable.

The main prototype page contains a file upload area, prediction results, and brief information about the diseases that can be recognized by the model. In the video-based scenario, the system extracts frames at an interval of approximately one frame per second. Each frame is then processed through RGB color conversion, resizing to 224×224 pixels, pixel normalization to the range of 0–1, data type conversion to float32, and batch

dimension expansion before being processed by the model.

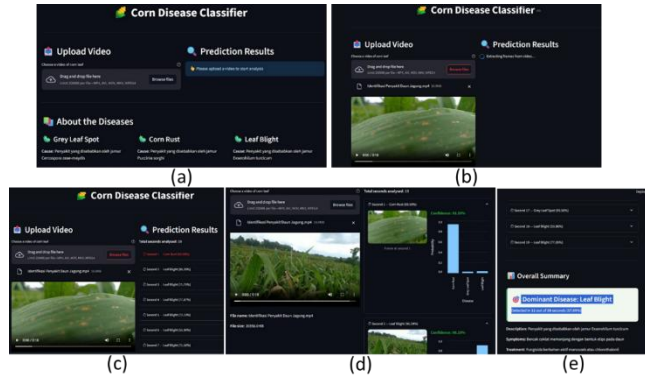


Fig. 7. Prototype interface: (a) main page, (b) frame extraction process from video, (c) frame analysis process using the model, (d) model classification result, and (e) overall summary display

The inference results are displayed as predicted classes and confidence values for each frame. After all frames are analyzed, the system generates an overall summary based on the frequency of predicted class occurrences. The class that appears most frequently is selected as the dominant prediction for the video.

Algorithm 1 Ekstraksi frame dari video yang diunggah

```

1: Input: File video  $V$ , frame rate cadangan  $f_0$ 
2: Output: Daftar frame hasil ekstraksi  $F$ 
3: Buka video  $V$ 
4: Baca frame rate aktual  $f_a$  dari  $V$ 
5: If  $f_a > 0$  then
6:    $step \leftarrow round(f_a)$ 
7: else
8:    $step \leftarrow f_0$ 
9: end if
10: Inisialisasi  $F \leftarrow \emptyset$ 
11: Inisialisasi indeks frame  $i \leftarrow 0$ 
12: Inisialisasi indeks detik  $t \leftarrow 0$ 
13: while video  $V$  masih dapat dibaca do
14:   Baca frame saat ini  $f_i$ 
15:   If  $f_i$  tidak valid then
16:     break
17:   end if
18:   If  $i \bmod step = 0$  then
19:     Konversi  $f_i$  dari BGR ke RGB
20:     Tambahkan  $(t, f_i)$  ke  $F$ 
21:      $t \leftarrow t + 1$ 
22:   end if
23:    $i \leftarrow i + 1$ 
24: end while
25: Lepaskan resource video
26: return  $F$ 

```

(a)

Algorithm 2 Klasifikasi penyakit dari jagung berbasis video menggunakan Corn

```

1: Input: Video dan jagung yang diunggah  $V$ , model ONNX  $M$ 
2: Output: Daftar prediksi tiap frame  $P$  dan penyakit dominan  $d$ 
3: If  $V$  tidak tersedia then
4:   Tampilkan pesan peringatan
5:   return proses
6: end if
7: Simpan  $V$  sebagai file sementara  $T$ 
8: Ekstrak frame  $F$  dari  $T$  dengan interval 1 frame per detik
9: If  $F$  kosong then
10:  Tampilkan pesan kesalahan
11:  return proses
12: end if
13: Inisialisasi daftar prediksi  $P \leftarrow \emptyset$ 
14: for all frame  $f_i \in F$  do
15:   Lakukan preproses  $f_i$  dengan mengubah ukurannya menjadi  $224 \times 224$  pik
16:   Normalisasi nilai piksel  $f_i$  ke rentang  $[0, 1]$ 
17:   Tambahkan dimensi batch pada tensor input
18:   Jalankan inferensi menggunakan model  $M$ 
19:   Peroleh vektor probabilitas  $p_i$ 
20:   Tentukan kelas prediksi  $c_i \leftarrow \arg\max(p_i)$ 
21:   Tentukan nilai confidence  $s_i \leftarrow \max(p_i)$ 
22:   Tambahkan  $c_i$  ke  $P$ 
23:   Tampilkan indeks detik, kelas prediksi, dan nilai confidence
24: end for
25: Tentukan penyakit dominan  $d$  dari  $P$  menggunakan perhitungan frekuensi
26: Hitung jumlah kemunculan  $d$  di dalam  $P$ 
27: Tampilkan ringkasan akhir penyakit dominan  $d$ 
28: Tampilkan deskripsi, gejala, dan penanganan dari  $d$ 
29: Hapus file sementara  $T$ 
30: return  $P, d$ 

```

(b)

Fig. 8. Algorithm implementation: (a) frame extraction and (b) video-based corn leaf disease classification using CornLeafNet

In addition to video-based classification, the prototype also supports single-image classification. In this scenario, the uploaded image is processed through the same preprocessing and inference stages. The prediction result is displayed together with the probability distribution and additional disease information. Therefore, the deployment prototype demonstrates that CornLeafNet can support corn leaf disease identification using both image and video inputs

3.5. Prototype Testing

Prototype testing was conducted using the Black Box Testing method [20]. This testing aimed to ensure that all core functions operated according to the expected design without examining the internal structure of the program code. The tested aspects included ONNX

model loading, image upload, video upload, preprocessing, inference, classification result display, and overall summary generation in the video-based scenario.

In the image-based scenario, the system was able to receive input images, display image previews, resize images to 224×224 pixels, normalize pixel values, and perform inference using the ONNX model. The system then displayed the predicted class, confidence value, and probability distribution.

In the video-based scenario, the system was able to receive video files, display the input video, automatically extract frames, preprocess each frame, perform classification, and display prediction results for each frame. The system was also able to summarize all frame predictions into a dominant disease class. In addition, the system could handle cases where no file had been uploaded by displaying an information message to the user. The summary of the prototype testing results is presented in Table 6.

Table 6. Summary of Prototype Testing Results

Testing Aspect	Expected Result	Status
ONNX model loading	The system successfully loads the ONNX model and creates an inference session	Successful
Corn leaf image upload	The system receives the image file and displays an input preview	Successful
Image preprocessing	The system performs resizing to 224×224 pixels and pixel value normalization	Successful
Image-based classification	The system displays the predicted class, confidence value, and probability distribution	Successful
Corn leaf video upload	The system receives the video file and displays the input video	Successful
Video frame extraction	The system extracts frames at an interval of approximately one frame per second	Successful
Per-frame classification	The system displays classification results and confidence values for each frame	Successful
Video overall summary	The system displays the dominant disease class from all analyzed frames	Successful
Interface and model integration	The prototype interface, preprocessing, ONNX inference, and system outputs run in an integrated manner	Successful

Based on the testing results, all core prototype functions were successfully executed. This indicates that the integration between the interface, image preprocessing, ONNX model inference, and classification result presentation operated according to the system design.

3.6. Discussion

The results of this study show that CornLeafNet is capable of classifying corn leaf diseases with high performance. The model's performance was influenced by several factors, including dataset quality, image condition variability, uniform input size, and pixel

normalization, which contributed to more consistent training and inference processes.

The disease-area-based approach through bounding box cropping played an important role in improving the model's learning focus. By using symptom regions as the main input, the model was able to learn visual patterns that were more relevant to disease characteristics, rather than background features that were not directly associated with symptoms. This represents an important distinction from previous studies that used whole-leaf images.

Furthermore, the use of a hierarchical CNN architecture equipped with batch normalization and dropout helped the model extract important features while reducing the risk of overfitting. Batch normalization supported training stability, while dropout reduced excessive dependence on specific neurons. This combination enabled CornLeafNet to produce strong classification performance on the testing dataset.

The successful conversion of the model into ONNX format also demonstrates that the model is not only effective during training but also suitable for deployment. The prediction difference between the TensorFlow and ONNX models, which approached 0.0000, indicates that the conversion process did not significantly alter the output characteristics of the model. Therefore, the ONNX model can be used appropriately as an inference component in the deployment prototype.

The prototype implementation provides additional practical value to this study. The system can classify not only single images but also video inputs through frame extraction. The overall summary feature in the video scenario enables users to obtain information about the dominant disease class from a sequence of analyzed frames. Thus, although the primary contribution of this study lies in the development of CornLeafNet as a disease-area-based CNN model, the deployment prototype demonstrates the feasibility of applying the model in practical identification scenarios.

Nevertheless, this study has several limitations. First, the model was developed only for three corn leaf disease classes: Grey Leaf Spot, Corn Rust, and Leaf Blight. Second, the proposed approach still depends on XML annotation and bounding box-based cropping. Third, video classification is still performed through frame extraction at specific intervals, meaning that the system does not yet fully utilize temporal video-based analysis. Therefore, future studies may focus on expanding disease classes, using more diverse datasets, developing end-to-end classification on whole-leaf images, optimizing inference performance, and implementing the model on broader platforms.

4. CONCLUSION

This study successfully developed CornLeafNet as a Custom Convolutional Neural Network model for disease-area-based corn leaf disease classification. The

proposed approach utilizes XML annotation and bounding box-based cropping to enable the model to focus on leaf regions that show disease symptoms. The model was developed to classify three disease classes, namely Grey Leaf Spot, Corn Rust, and Leaf Blight.

The training results show that CornLeafNet achieved a validation accuracy of 97.66% and a validation loss of 0.0663. On the testing dataset, the model achieved an accuracy of 98.60%, a loss of 0.0450, and precision, recall, and F1-score values of 0.9860, respectively. These results indicate that CornLeafNet can effectively extract visual features of corn leaf diseases and produce high classification performance.

The best-performing model was successfully converted into ONNX format and deployed in a web-based prototype. The prototype supports two input scenarios: image and video. For image input, the system displays the prediction result, confidence value, and probability distribution. For video input, the system extracts frames, performs classification on each frame, and generates an overall summary in the form of the dominant disease class.

Overall, CornLeafNet has the potential to serve as an automatic, rapid, and practical support model for early corn leaf disease identification. However, this study remains limited to three disease classes, XML annotation-based data, and frame extraction-based video analysis. Future research should focus on dataset expansion, additional disease classes, end-to-end classification using whole-leaf images, inference performance optimization, and implementation on wider platforms

ACKNOWLEDGEMENTS

The authors would like to thank all parties who provided support and assistance during the completion of this research.

REFERENCES

- [1] F. Melia, F. M. Aldian, M. S. F. Pahlevi, R. N. I. Risquallah, and S. Oktaffiani, "PERAN PEMERINTAH DALAM MENINGKATKAN VOLUME EKSPOR JAGUNG," *JURNAL ECONOMINA*, vol. 2, no. 1, pp. 1305–1320, Jan. 2023, doi: 10.55681/economina.v2i1.287.
- [2] P. S. Kobesi, H. Firmansyah, W. B. Devita, O. W. Matoneng, and A. Sarifuddin, "STRATEGI PENGEMBANGAN USAHATANI JAGUNG DI KABUPATEN TANAH LAUT, PROVINSI KALIMANTAN SELATAN," *ZIRAA'AH MAJALAH ILMIAH PERTANIAN*, vol. 51, no. 2, p. 607, May 2026, doi: 10.31602/zmip.v51i2.22922.
- [3] A. P. N. Kamal, A. Suyadi, T. Pribadi, and O. D. Hajoeningtjas, "Pengaruh Defoliiasi dan Pemberian Pupuk MKP terhadap Pertumbuhan dan Hasil Jagung (*Zea Mays L.*)," *Proceedings Series on Physical & Formal Sciences*, vol. 8, pp. 36–41, Mar. 2025, doi: 10.30595/pspfs.v8i.1469.
- [4] Prasetyo Mimboro, "Identifikasi Penyakit Tanaman Jagung berdasarkan Citra Daun Menggunakan Hybrid k-NN - CNN,"

- Journal of Cyber Health and Computer*, vol. 1, no. 1, pp. 6–9, Dec. 2023, doi: 10.64163/jochac.v1i1.2.
- [5] H. Hamidson and R. A. Efendi, "The Main Disease and its Attacks in the Generative Phase of Maize (*Zea Mays* L) in the Freshwater Swamps of South Sumatra," *Jurnal Lahan Suboptimal: Journal of Suboptimal Lands*, vol. 10, no. 2, pp. 195–201, Oct. 2021, doi: 10.36706/jlso.10.2.2021.540.
- [6] D. S. Sainong, R. Iswati, N. Nurdin, A. P. Solihin, S. H. Purnomo, and S. I. Pulogu, "Evaluasi Efektivitas Aplikasi Kompos *Trichoderma reesei* (Isolat TZ31DU1) Sebagai Agen Hayati dalam Menekan Serangan Penyakit Penting pada Tanaman Jagung," *AGROSCRIPT: Journal of Applied Agricultural Sciences*, vol. 7, no. 1, pp. 109–125, Jun. 2025, doi: 10.36423/agroscript.v7i1.2244.
- [7] M. Martinez-Ramon, M. Ajith, and A. R. Kurup, *Deep Learning*. Wiley, 2024. doi: 10.1002/9781119861898.
- [8] H. Habibi Aghdam and E. Jahani Heravi, *Guide to Convolutional Neural Networks*. Cham: Springer International Publishing, 2017. doi: 10.1007/978-3-319-57550-6.
- [9] Badan Pusat Statistik Provinsi Nusa Tenggara Barat, "Luas Panen dan Produksi Jagung di Provinsi Nusa Tenggara Barat 2024 (Angka Tetap)," Mar. 2025. [Online]. Available: <https://ntb.bps.go.id/id/pressrelease/2025/03/03/1098/luas-panen-dan-produksi-jagung-di-provinsi-nusa-tenggara-barat-2024-angka-tetap-.html>
- [10] Pemerintah Provinsi Nusa Tenggara Barat, "Panen Jagung Bersama, Mentan RI Optimis Gubernur NTB Mampu Jadikan Sumbawa sebagai Sentra Hilirisasi Pangan," Apr. 2025. [Online]. Available: <https://ntbprov.go.id/post/panen-jagung-bersama-mentan-ri-optimis-gubernur-ntb-mampu-jadikan-sumbawa-sebagai-sentra-hilirisasi-pangan>
- [11] U. Khaira, I. Weni, and W. Wilia, "Rancang Bangun Aplikasi Deteksi Penyakit Tanaman Jagung Melalui Citra Daun Berbasis Android Menggunakan Algoritma Convolutional Neural Network," *Jurnal Pepadun*, vol. 5, no. 1, pp. 1–11, Apr. 2024, doi: 10.23960/pepadun.v5i1.210.
- [12] Muhammad Naufal Anugrah and Nur Rachmat, "Klasifikasi Penyakit Daun Tanaman Jagung Menggunakan Pendekatan Transfer Learning Arsitektur MobileNetV4," *Jurnal Teknologi dan Manajemen Industri Terapan*, vol. 4, no. 4, pp. 2174–2182, Dec. 2025, doi: 10.55826/jtmit.v4i4.1393.
- [13] M. Rijal, A. M. Yani, and A. Rahman, "Deteksi Citra Daun untuk Klasifikasi Penyakit Padi menggunakan Pendekatan Deep Learning dengan Model CNN," *Jurnal Teknologi Terpadu*, vol. 10, no. 1, pp. 56–62, Jul. 2024, doi: 10.54914/jtt.v10i1.1224.
- [14] Kusdatul Komariyah, Ratri Enggar Pawening, and Moh Furqan, "Klasifikasi Penyakit Jagung Berdasarkan Citra Batang Menggunakan Metode Convolutional Neural Network (CNN) pada Arsitektur Mobilenet Berbasis Web," *JOKI: Journal of Computing and Informatics*, vol. 2, no. 2, pp. 11–19, Dec. 2025, doi: 10.65678/joki.v2i2.206.
- [15] I. Wirabowo and I. Susilawati, "Implementasi Convolution Neural Network (CNN) untuk Deteksi Penyakit pada Daun Jagung Berbasis Citra Digital," *Jurnal Pustaka Data (Pusat Akses Kajian Database, Analisa Teknologi, dan Arsitektur Komputer)*, vol. 5, no. 1, pp. 233–241, Jun. 2025, doi: 10.55382/jurnalpustakadata.v5i1.1046.
- [16] H. Herfandi, R. Bin Mofidul, and I. A. Khan, "Structural Correlation Patterns in Regional COVID-19 Surveillance Data and Implications for Epidemiological Monitoring," *Jurnal Masyarakat Informatika*, vol. 17, no. 1, pp. 113–124, Feb. 2026, doi: 10.14710/jmasif.17.1.77361.
- [17] H. Herfandi, M. A. Aldiansyah, M. J. M. Julkarnain, and A. Susilo Yuda Irawan, "Penerapan Algoritma You Only Look Once (YOLO) Versi 8 dan Convolutional Neural Network (CNN) Dan Image Processing pada Aplikasi Automatic Number Plate Recognition (ANPR)," *Syntax: Jurnal Informatika*, vol. 14, no. 01, pp. 63–77, May 2025, doi: 10.35706/syji.v14i01.13074.
- [18] H. Herfandi *et al.*, "Hybrid OOK-OFDM Optical Camera Communication for Real-Time Indoor Mobility With Multi-LED Support," *IEEE Internet Things J.*, vol. 13, no. 5, pp. 8548–8560, Mar. 2026, doi: 10.1109/JIOT.2025.3641627.
- [19] H. Herfandi, S. Dwiasnati, K. A. Baihaqi, and R. Avrizal, "Perancangan Sistem Penerimaan Siswa Baru Berbasis Web Pada Sekolah Dasar Islam Plus Baitul Maal," *Faktor Exacta*, vol. 15, no. 2, Aug. 2022, doi: 10.30998/faktorexacta.v15i2.12894.
- [20] H. Herfandi, U. Soleha, A. S. Y. Irawan, K. A. Baihaqi, and R. Maulana, "Implementasi Algoritma Best First Search untuk Pencarian Rute Terpendek pada Aplikasi Cerdas Pendaftaran Santri Baru," *Syntax: Jurnal Informatika*, vol. 11, no. 1, pp. 63–74, 2022.