Jurnal CoreIT, Vol.11, No.2, December 2025
ISSN 2460-738X (Print)
ISSN 2599-3321 (Online)

Classification of Apple Tree Leaf Diseases using Pretrained EfficientNetB0 and XGBoost

Bagus Al Qohar[1*], Ahmad Ubai Dullah[1], Aditya Yoga Darmawan[1], Jumanto Unjung[1]
[1]Dept. of Computer Science, Universitas Negeri Semarang, Indonesia
bagusximipa6@students.unnes.ac.id*, ubaid@students.unnes.ac.id, darmoenoyoga@students.unnes.ac.id, jumanto@mail.unnes.ac.id

Abstract. The diseases that affect apple tree leaves seriously compromise agricultural production; therefore, early and accurate diagnosis is crucial for effective disease control. Machine learning's recent developments have opened up fascinating possibilities for automating the detection process and enhancing precision agriculture methods. This study aims to develop a robust classification model that can accurately and efficiently identify various diseases affecting apple tree leaves. The approach combines the pre-trained EfficientNetB0 architecture for feature extraction with the XGBoost model for classification, utilizing the advantages of both deep learning and gradient-boosting methods. With high performance measures including a macro-average precision of 95.86%, recall of 95.44%, and F1 score of 95.64%, the model achieved a classification accuracy of 95.74%. Furthermore, the average ROC-AUC score of 0.9964 highlights the model's ability to effectively differentiate between the five disease categories. This work stands out due to its hybrid approach, which integrates a robust pre-trained convolutional neural network (EfficientNetB0) with the XGBoost model. This significantly improves the accuracy of disease classification. This approach presents a novel pathway for precision agriculture, providing a reliable and effective instrument for the automatic identification of diseases in apple orchards.

Keywords: Apple Leaf Diseases, Classification Model, EfficientNetB0, Machine Learning, XGBoost.

Received December 2024/ **Revised** August 2025/ **Accepted** December 2025

INTRODUCTION

The direct effect of apple leaf diseases on crop production and quality makes their identification a vital focus of study. Automated solutions are driven by conventional disease detection approaches, which depend on manual inspection and are often error-prone and labor-intensive. Recent developments in machine learning (ML) and deep learning (DL) have created new paths for highly accurate and efficient leaf disease detection. For example, [1] emphasized the need for ML to increase the accuracy of disease identification by offering a comprehensive review of apple leaf disease detection methods [2]. Similarly, [3] predicted diseases in apple leaves using multiclass support vector machines (SVMs). With more effective feature extraction, they achieved a significantly improved classification. Researchers have investigated the junction of DL models and machine learning classifiers in order to improve the detection accuracy even more. In one study, [4] proposed a deep learning framework to detect apple leaf diseases using image processing methods combined with convolutional neural networks (CNNs) for feature extraction [5]. Their technique demonstrated notable improvements in accuracy and precision compared to conventional techniques.

Research by [6] discussed how well pre-trained networks, such as EfficientNet, find complex disease patterns and examined new advances in deep learning approaches for disease detection on apple leaves. Their study highlighted that more accurate and computationally efficient models are generated when EfficientNet is combined with machine learning classifiers, such as XGBoost. This trend is especially pertinent given the growing attention to hybrid models that combine DL architectures with conventional ML classifiers for best performance. To search for diseases in apple leaves, [7] developed MCDCNet, a multiscale limited deformable convolution network. It outperformed many baseline models in terms of accuracy, precision, and recall, and effectively captured elements in space on varying scales. To greatly increase detection accuracy, [8] also developed a deep evidence fusion framework for apple leaf disease classification by including several data sources. These investigations highlight how well multiscale evidence fusion techniques could improve the detection process.

Focusing on improving the robustness and generalization of the model, recent studies have continued to investigate advanced approaches for detecting apple leaf disease. [9] investigated various deep learning and

DOI: 10.24014/coreit.v11i2.33174

machine learning techniques extensively for detecting diseases on apple leaves. They found that the most successful hybrid models were those that combined the best features of both approaches. Emphasizing the importance of feature extraction to enhance model precision, [10] also examined the effectiveness of machine learning and deep learning methods for classifying apple leaf disease. [11] conducted the last work classifying lepidoptera using EfficientNet-B0. This indicated that it could support the possible use in identifying and extracting characteristics for classification activities. These works validate the trend of using hybrid models and pre-trained networks for enhanced disease detection in precision agriculture.

The uniqueness of hybridizing machine learning classifiers, such as XGBoost, with deep learning architectures, including EfficientNet and MCDCNet, lies in increasing both accuracy and generalizability. Through adaptive feature capture approaches, researchers such as [12] and [13] have focused on enhancing segmentation and detection, underscoring the potential of these cutting-edge methods to continually improve classification performance. However, existing studies have not fully addressed the challenge of distinguishing visually similar leaf diseases under real-world variations. To bridge this gap, this research addresses these challenges by developing a hybrid classification framework that integrates a pretrained EfficientNetB0 model for deep feature extraction with the XGBoost classifier to improve decision boundaries. The novelty of this research leverages the unique properties of the EfficientNetB0 model, which serves as a powerful yet lightweight backbone, and the adaptive ensemble characteristics of XGBoost. Consequently, the deeper neural network achieves better performance on limited or imbalanced datasets. The hybrid model improved the generalization and classification of apple tree leaf diseases.

METHODS

The proposed method for classifying apple tree leaf diseases uses the strengths of both deep learning and machine learning approaches [14], [15]. This paper utilizes EfficientNetB0, a compact yet highly effective convolutional neural network (CNN), to extract features. Pre-trained weights from the ImageNet dataset is used to transfer its learning abilities. This paper also extracts deep feature representations from apple leaf images and subsequently feeds these features to an XGBoost model, a gradient boosting method known for great accuracy and robustness in classification problems. EfficientNetB0's feature extraction, combined with XGBoost's classification, offers a hybrid model that successfully strikes a balance between computational efficiency and performance accuracy [11]. Using criteria including precision, recall, F1 score, and ROC-AUC, this article evaluated the model's performance and found notable improvements in the identification of several apple leaf diseases. Figure 1 shows a method flowchart diagram.

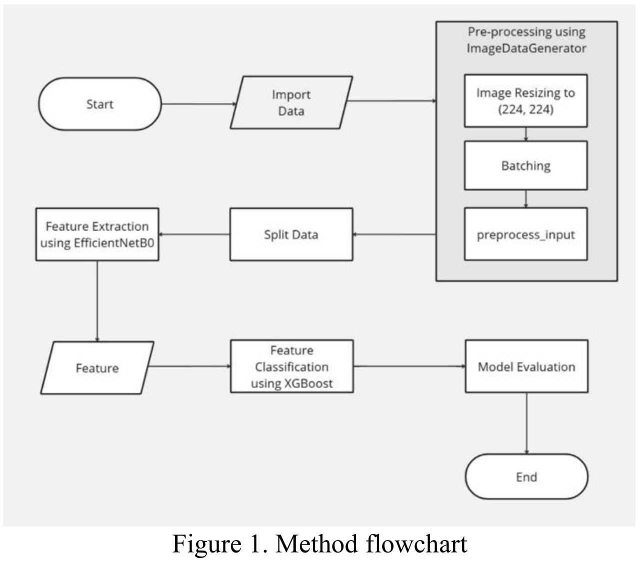
Start
Import Data
Pre-processing using ImageDataGenerator
Image Resizing to (224, 224)
Batching
preprocess_input
Split Data
Feature Extraction using EfficientNetB0
Feature
Feature Classification using XGBoost
Model Evaluation
End

Figure 1. Method flowchart

A. Import Data

In this research, the dataset used consists of an apple tree leaf disease dataset from five different categories. These diseases include alternating leaf spot, gray spot, brown spot, and rust. The data are collected from the Kaggle website [16], and the overall number of images is 1641. The data is then divided into separate training and testing sets. The distribution of data for each class is shown in Table 1.

Table 1. Apple tree leaf disease dataset distribution

Class	**Number of Images**
Alternia Leaf Spot	278
Brown Spot	215
Gray Spot	395
Healthy Leaf	409
Rust	344
Total	1641

B. Images Preprocessing

The preprocessing of the dataset ensures an optimal structure for the training and testing of the machine learning models in this work. The preprocessing starts with creating training and testing sets from the data. As shown in the code, the split data function divides the images from the source directory into separate directories for training and testing, thereby automating this process. Using the split ratio parameter allows one to vary the allocation of images, with 80% allocated to the training set and the remaining 20% to the testing set. Every class of a picture is arranged in line with its related directory.

The TensorFlow Keras API initiates preprocessing as soon as the data is split. The pre-process input function of the image data generator class establishes a standard pixel value for images. Deep learning relies on standardization, as it ensures the appropriate scaling of input data, enhances model convergence, and yields better results. This is especially important when using pre-trained models, such as InceptionV3, where the input must conform to a specific structure.

From the training and test directories, the Image Data Generator then generates batches of images [17]. The images are loaded straight from their directories, resized to the suitable input shape of 224x224 pixels, and created batches of a specified size, in this case 32 images per batch using the flow from directory method. Setting class mode to None and shuffle to False helps the generator prepare images without labels, which can subsequently be used for classification chores once the model is fully trained. Figure 2 and Figure 3 show the different images before and after pre-processing.

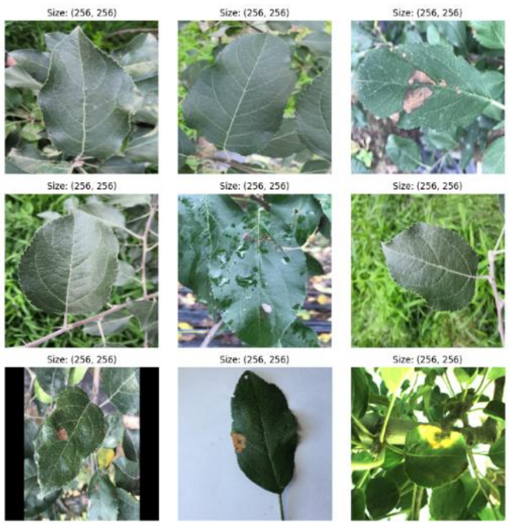

Figure 2. Sample of images before preprocessing

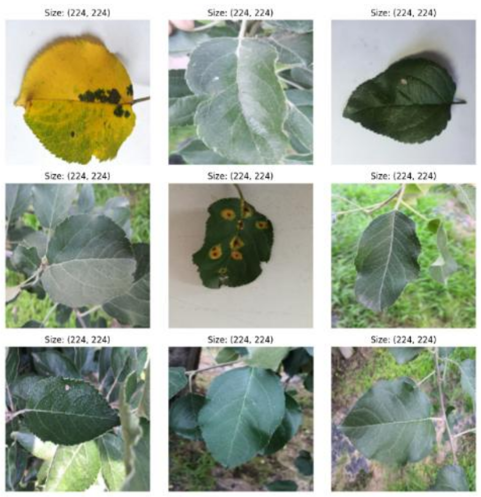

Figure 3. Sample of images after preprocessing

C. Feature Extraction

In this work, for feature extraction, the ImageNet dataset is pre-trained using EfficientNetB0, a powerful convolutional neural network (CNN) architecture [18], [19]. EfficientNetB0's effective scaling of depth, width, and resolution has shown outstanding performance in many image classification challenges. This paper seeks to extract high-quality features from our apple tree leaf dataset using pre-trained weights, avoiding the need to build the whole model from scratch. It is perfect for feature extraction, since the architecture is used without a top classification layer. Figure 4 shows a summary of the EfficientNetB0 architecture model.

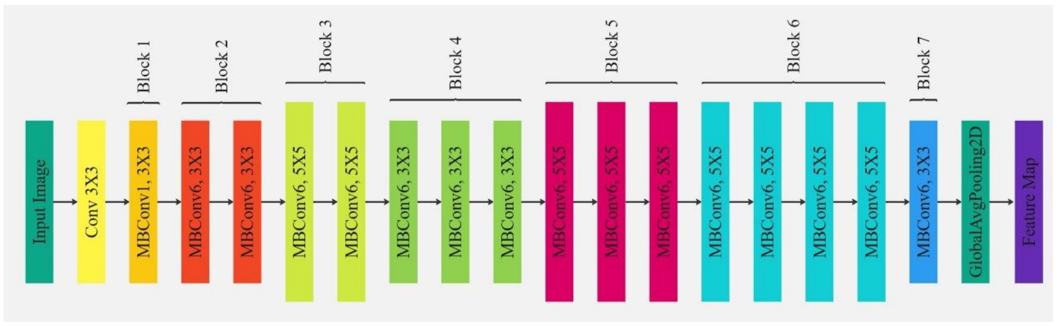

Figure 4. EfficientNetB0 architecture model

Following the base model, this paper adds a Global Average Pooling 2D (GAP) layer to further hone the EfficientNetB0 extracted features. By reducing the spatial dimensions of the output feature maps, the GAP layer substitutes for conventional fully connected layers and retains the most significant information. By concatenating the feature maps into a single value per feature, this layer reduces the general complexity of the feature vectors. Using GAP produces compact and globally representative features from the input images that fit downstream applications, including classification. Figure 4 shows model architecture of EfficientNetB0.

Feature extraction is applied with a Keras model that combines the GAP layer with EfficientNetB0. The raw images form the model's input; the feature vectors produced following the GAP layer form its output. This paper uses a custom function, extract_features, to effectively manage this process. Using the pre-trained EfficientNetB0 model, this function iteratively over ImageDataGenerator's produced image batches forecasts the feature vectors. Dividing the total sample count by the batch size helps us determine the number of steps needed to handle the dataset. To ensure that the model generates significant feature vectors for both the training and testing sets, feature vectors are also extracted for both sets. The main input for additional analysis or training of the machine learning pipeline is these derived features. For every dataset, the function outputs feature arrays that allow us to examine the dimensions and understand the scale and organization of the data obtained. For instance, the form of the feature arrays reveals the number of feature vectors generated for every image, which is a crucial insight for the next model construction.

D. Feature Classification

In this work, the obtained features are classified using the XGBoost model. Particularly in large datasets, XGBoost is a strong and very efficient implementation of gradient boosting that shines in speed and accuracy [20], [21], [22]. XGBoost generates a strong classifier capable of fast differentiating between the five classes of apple tree leaf images by using pre-extracted feature vectors as input. The model is trained with the hyperparameter method shown in Table 2.

Table 2. Hyperparameter for classification

Hyperparameter	Value	Description
n_estimators	500	Number of trees in the ensemble (boosting rounds)
max_depth	5	Maximum depth of each tree. Controls the complexity of the model.
learning_rate	0.1	Shrinks the contribution of each tree to prevent overfitting.
random_state	42	Seed for random number generation to ensure reproducibility
device	GPU	Specifies the use of GPU for faster training

The XGBoost classifier learns from the features acquired for training. The classifier receives training from the feature vectors of the training set train_features, even if the class labels from train_gen.classes do not match. Following training, the classifier projects the test set (test_features) relative to the actual class labels of the test data. Evaluating the performance of the model depends on this predictive phase since it helps us to determine how well the classifier generalizes to unprocessed data.

This paper evaluates the classifier's performance by computing the accuracy of the model on the test set by means of the accuracy_score function and presents a rather high percentage accuracy of the model to provide a complete picture of its classification performance. In addition, the tool generates comprehensive classification reports. This article presents key metrics, including precision, recall, F1 score, and class support. These steps assess how well the model handles imbalanced classes and whether it performs consistently across all dataset categories, providing a more comprehensive understanding of its performance.

E. Model Evaluation

To obtain a more comprehensive evaluation, additional metrics, such as precision, recall, and the F1 score, are calculated using the model's predictions [23], [24]. Precision measures the ratio of correct positive predictions to total positive predictions, which highlights the accuracy of the model in predicting a particular class. Recall, on the other hand, estimates a model based on its ability to identify all relevant examples in the validation set, thereby underlining the sensitivity regarding real positives. The F1 score is the harmonic mean of precision and recall. Balances the two measures and provides a more detailed view of model performance when one is less important. Combining precision, recall, and F1-score ensures the completeness of judgment on a model's effectiveness.

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

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

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

$$F1\,Score = 2 \times \frac{recall \times precision}{recall + precision} \quad (4)$$

$$TPR = \frac{TP}{TP+FN} \quad (5)$$

$$FPR = \frac{FP}{FP+TN} \quad (6)$$

Where TP is True Positive, where the model correctly predicts the positive class. TN is True Negative, where the model correctly predicts the negative class. FP is False Positive, where the model incorrectly predicts the positive class when the actual class is negative, FN is False Negative, the model incorrectly predicts the negative class when the actual class is positive. TPR and FPR are used to plot the Receiver Operating Characteristic (ROC).

RESULT AND DISCUSSION

A. Feature Extraction Result

Using the pre-trained EfficientNetB0 model for feature extraction, the raw image data were successfully converted into a format that the XGBoost classifier could understand. The results are presented in Table 3.

Table 3. Feature extraction

	Train	Test
Shape	1,312	329
Feature	1,280	1280

Based on Table 3, the output indicates the conversion of the training set, comprising 1,312 images, into a feature matrix with dimensions of 1,312 rows and 1,280 columns. Similarly, a feature matrix with 329 rows and 1,280 columns now represents the test set of 329 images. The EfficientNetB0 model extracts distinct features from each of the 1,280 columns, and each row in these matrices corresponds to an image.

The EfficientNetB0 model learned high-level visual information from these 1,280 features during its training on a massive image dataset, likely ImageNet. Instead of using raw pixel values, which can be high-dimensional and noisy, this paper leverages these learned features that capture more discriminative and abstract image representations. The pre-trained models can acquire knowledge for our specific image classification task through this approach, known as transfer learning.

A key turning point in our image classification system is the effective extraction of these 1280-dimensional feature vectors from the image datasets. The XGBoost classifier will learn to map these features to related image labels using these feature matrices as input. A strong and accurate image classification system can be achieved by combining deep learning for feature extraction with a robust machine learning algorithm, such as XGBoost.

B. Feature Classification Result

Training on features derived from the EfficientNetB0 model, the XGBoost model has shown extraordinary image classification performance. With an impressive accuracy of 95.74%, the classifier accurately predicted the class labels for most images in the test set. This exceptional accuracy demonstrates the effectiveness of transfer learning in combination with a robust machine learning method for image recognition.

Examining the classification report closely repeatedly reveals high precision, recall, and F1 scores in all classes, with each score above 0.9. This indicates a well-balanced performance in which the classifier not only detects most of the instances matching each class (recall), but also finds the proper class with high confidence (precision). Harmonizing accuracy and recall, the F1 score strengthens the classifier's general performance across all classes.

Many factors contribute to explaining this exceptional performance. Firstly, the extraction of features with EfficientNetB0 ensured the discriminative and informative components of the XGBoost model. Second, well known for its ability to handle complex datasets and find subtle relationships, the XGBoost algorithm effectively learned the mapping between the extracted features and the matching class labels. The classifier's exceptional performance ultimately stems from the meticulous selection of hyperparameters and the utilization of GPU devices. Table 4 presents a comprehensive analysis of the classification outcomes for each class.

Table 4. Feature classification result

Class	Precision	Recall	F1-Score	Support
0	0.963	0.9286	0.9455	56
1	0.9302	0.9302	0.9302	43
2	0.9146	0.9494	0.9317	79
3	0.9878	0.9878	0.9878	82
4	0.9853	0.971	0.9781	69
Accuracy				329
macro avg	0.9562	0.9534	0.9547	329
weighted avg	0.958	0.9574	0.9576	329

Details:
Class 0: Alternaria leaf spot
Class 1: Brown spot
Class 2: Gray spot
Class 3: Healthy leaf
Class 4: Rust

C. Confusion Matrix

A confusion matrix is a common tool used to evaluate the performance of a classification model, specifically one designed to classify different types of data. Figure 5 shows a confusion matrix. The matrix compares the predicted types of apple tree leaf disease with the actual types of apple tree leaf disease. A variety of blue hues fill the cells, and darker blues represent the frequency of correct or incorrect predictions. This color gradient makes it easier to visualize the model's accuracy and errors.

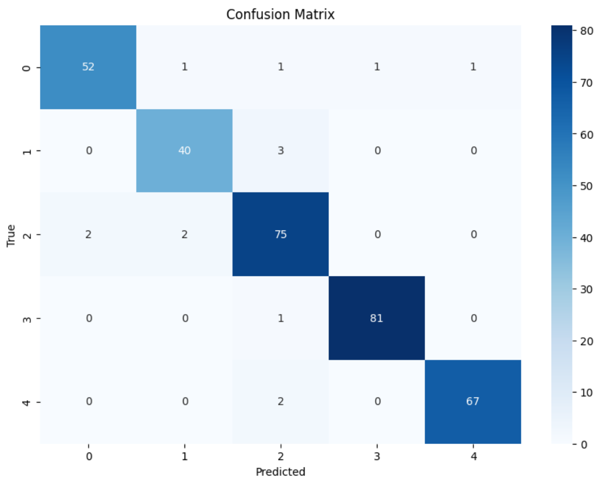

Figure 5. Confusion matrix

Figure 5 presents the five distinct classes that a classification model can predict for apple tree leaf disease. This paper sets up the matrix as a 5x5 grid, where each column represents the predicted class, and each row represents the true class. The model correctly classified 52 samples in class 0, 40 in class 1, 75 in class 2, 81 in class 3, and 67 in class 4, with diagonal values of 52, 40, 75, 81, and 67 for each class, respectively. Examining off-diagonal values reveals misclassifications. In class 0, for example, the model misclassified one sample as class 1, another as class 2, another as class 3, and still another as class 4. In class 1, the model also incorrectly predicted three samples, as it did in class 2. In class 2, the model misclassified two samples as class 0 and another two as class 1. For class 3, the model misclassified only one sample, projecting it as class 2. Finally, class 4 misinterpreted two samples as class 2.

Based on the higher diagonal values and relatively lower misclassification rates, the model demonstrated excellent general performance for most classes. With 81, Class 3 boasts the most accurate predictions; Category 1 has the fewest misclassifications, with only 3 incorrect predictions. With a minor uncertainty between similar classes, such as between classes 1 and 2, and between classes 2 and 4, misclassifications appear to be rare, and the model is able to precisely differentiate between the various disease categories.

This implies that diseases in these categories may share similar visual traits; therefore, the model would find it more challenging to distinguish them.

D. AUC-ROC Analysis

Figure 6 shows the Receiver Operating Characteristic (ROC) curve, which illustrates the ability of the model to differentiate between the five categories of apple leaf disease. Plotting the ROC curves for every class allows one to see the trade-off between the false positive rate across many threshold values and the true positive rate (sensitivity).

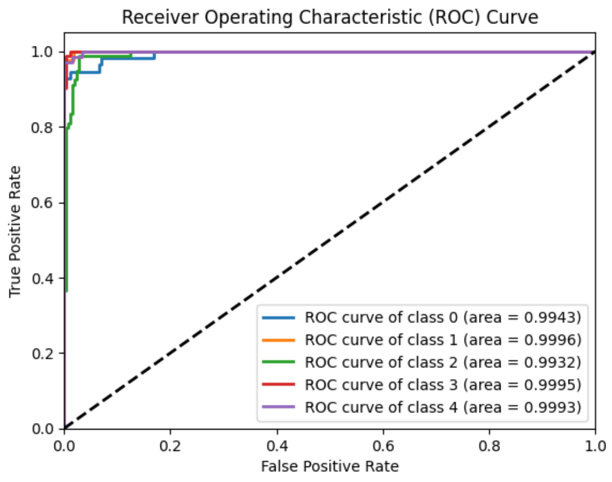

Figure 6. AUC-ROC curve

The Area Under the Curve (AUC) values for each class are especially notable, with class 0 achieving 99.43%, class 1 achieving 99.96%, class 2 achieving 99.32%, class 3 achieving 99.95%, and class 4 achieving 99.93%. With almost flawless discrimination between classes, these high AUC values indicate excellent model performance. The model's efficiency in differentiating between apple leaf diseases is shown by the closely approaching top-left corner's high true positive rates with low false positives.

E. Discussion

Table 5 compares the findings of this work with those of earlier studies on the classification of apple tree leaf diseases. Specifically, the proposed model, which integrates XGBoost model for classification and EfficientNetB0 for feature extraction, excels across all evaluation criteria. The model performs exceptionally well in confusion matrix evaluations such as precision, recall, and F1-score, demonstrating its ability to accurately classify a variety of apple leaf diseases, including Alternaria leaf spot, Brown spot, Gray spot, Healthy Leaf, and Rust. The model achieves a high AUC-ROC score of 99.7%.

Table 5. Comparison with previous study

Study	Dataset	Objective	Method	Result
[3]	• The dataset includes 500 apple leaf images.	To detect diseases in apple leaves using machine learning	Multiclass SVM	Multiclass SVM achieves 96% accuracy in identifying diseases.
[4]	• The dataset is collected from the Kaggle repository. • Apple leaf images are used for classification. • The dataset is split into training and testing sets.	To diagnose apple leaf diseases accurately using deep learning	VGG16	The VGG16 framework achieves 93.3% validation accuracy in the apple leaf data set.

[6]	• The dataset is sourced from the Cornell Initiative for Digital Agriculture. • It consists of around 3600 labeled training images.	To enhance apple leaf disease detection accuracy using deep learning	EfficientNet-DenseNet	The proposed method outperformed the others with 98% accuracy.
[8]	• The AppleLeaf9 dataset includes 14,582 images of 8 disease classes. • Dataset from Northwest AF University contains 26,377 apple leaf images.	To achieve high accuracy in leaf disease classification.	EfficientNetV2-S	The experimental results show improved classification accuracy, achieving 98.1% in EfficientNetV2-S.
Proposed Method	• The study uses a dataset of 1641 apple leaf images. • The dataset is collected from the Kaggle repository.	To create a strong classification model for apple leaf diseases.	EfficientNetB0 for feature extraction and XGBoost model for classification.	Achieved 95.74% classification accuracy and 0.9964 ROC AUC score.

CONCLUSION

This paper demonstrated the effectiveness of a hybrid model that combines the best features of the XGBoost model and a pre-trained EfficientNetB0 CNN in classifying apple tree leaf diseases. The model achieved 95.74% classification accuracy, 95.64% F1-score, and 95.64% macro-average precision by extracting features using EfficientNetB0 and XGBoost. The model's average ROC AUC score of 0.9964 accurately distinguishes the five disease categories. These findings suggest that hybrid models, which combine deep learning feature extraction with machine learning classifiers, may facilitate the identification of crop diseases. This study utilizes a pre-trained CNN model and an XGBoost classifier to detect agricultural diseases. This method enables farmers and agricultural experts to identify and control apple leaf diseases by enhancing computational efficiency and improving classification accuracy. Further research can examine other pretrained models, expand the dataset, or test this hybrid approach on different crops and diseases to prove its adaptability.

REFERENCES

[1] S. Bashir, F. Firdous, and S. Z. Rufai, "A Comprehensive Review on Apple Leaf Disease Detection," in *2023 IEEE 8th International Conference for Convergence in Technology (I2CT)*, IEEE, Apr. 2023, pp. 1–6. doi: 10.1109/I2CT57861.2023.10126487.

[2] V. K. Vishnoi, K. Kumar, B. Kumar, S. Mohan, and A. A. Khan, "Detection of Apple Plant Diseases Using Leaf Images Through Convolutional Neural Network," *IEEE Access*, vol. 11, pp. 6594–6609, 2023, doi: 10.1109/ACCESS.2022.3232917.

[3] S. Chakraborty, S. Paul, and M. Rahat-uz-Zaman, "Prediction of Apple Leaf Diseases Using Multiclass Support Vector Machine," in *2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, IEEE, Jan. 2021, pp. 147–151. doi: 10.1109/ICREST51555.2021.9331132.

[4] S. K, V. R. P, R. P, P. K. M, and P. S, "Apple Leaf Disease Detection using Deep Learning," in *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, IEEE, Mar. 2022, pp. 1063–1067. doi: 10.1109/ICCMC53470.2022.9753985.

[5] Y. Zhong and M. Zhao, "Research on deep learning in apple leaf disease recognition," *Comput. Electron. Agric.*, vol. 168, p. 105146, Jan. 2020, doi: 10.1016/j.compag.2019.105146.

[6] E. Kannan, C. M. B. M J, A. D. S, R. N. N, A. Begum, and H. D, "Deep Learning Techniques Advancements in Apple Leaf Disease Detection," *Procedia Comput. Sci.*, vol. 235, pp. 713–722, 2024, doi: 10.1016/j.procs.2024.04.068.

[7] B. Liu, X. Huang, L. Sun, X. Wei, Z. Ji, and H. Zhang, "MCDCNet: Multi-scale constrained deformable convolution network for apple leaf disease detection," *Comput. Electron. Agric.*, vol. 222, p. 109028, Jul. 2024, doi: 10.1016/j.compag.2024.109028.

[8] H. Wang, J. Zhang, Z. Yin, L. Huang, J. Wang, and X. Ma, "A deep evidence fusion framework for apple leaf disease classification," *Eng. Appl. Artif. Intell.*, vol. 136, p. 109011, Oct. 2024, doi: 10.1016/j.engappai.2024.109011.

[9] B. U. Rani, K. Pavani, S. Bhavani, and G. Alapati, "A Systematic Analysis of Deep Learning and Machine Learning Methods for Identifying Apple Leaf Disease," in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, IEEE, Jul. 2023, pp. 761–765. doi: 10.1109/ICESC57686.2023.10192948.

[10] M. Sebastian, S. M S, and C. M. Antony, "Apple Leaf Disease Detection: Machine Learning &amp; Deep Learning Techniques," in *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*, IEEE, Dec. 2023, pp. 1–5. doi: 10.1109/ICCEBS58601.2023.10449037.

[11] H. Syamsudin, S. Khalidah, and J. Unjung, "Lepidoptera Classification Using Convolutional Neural Network EfficientNet-B0," *Indones. J. Artif. Intell. Data Min.*, vol. 7, no. 1, p. 47, Nov. 2023, doi: 10.24014/ijaidm.v7i1.24586.

[12] S. Zhu, W. Ma, J. Lu, B. Ren, C. Wang, and J. Wang, "A novel approach for apple leaf disease image segmentation in complex scenes based on two-stage DeepLabv3+ with adaptive loss," *Comput. Electron. Agric.*, vol. 204, p. 107539, Jan. 2023, doi: 10.1016/j.compag.2022.107539.

[13] C. Yan and K. Yang, "FSM-YOLO: Apple leaf disease detection network based on adaptive feature capture and spatial context awareness," *Digit. Signal Process.*, vol. 155, p. 104770, Dec. 2024, doi: 10.1016/j.dsp.2024.104770.

[14] O. H. Kesav and R. G. K, "A Systematic Study on Enhanced Deep Learning Based Methodologies for Detection and Classification of Early Stage Cancers," in *2023 IEEE 5th International Conference on Cybernetics, Cognition and Machine Learning Applications (ICCCMLA)*, IEEE, Oct. 2023, pp. 328–333. doi: 10.1109/ICCCMLA58983.2023.10346973.

[15] S. Qian, T. Peng, Z. Tao, X. Li, M. S. Nazir, and C. Zhang, "An evolutionary deep learning model based on XGBoost feature selection and Gaussian data augmentation for AQI prediction," *Process Saf. Environ. Prot.*, vol. 191, pp. 836–851, Nov. 2024, doi: 10.1016/j.psep.2024.08.119.

[16] Kaggle, "Apple Tree Leaf Disease Dataset," Kaggle. Accessed: Oct. 02, 2024. [Online]. Available: https://www.kaggle.com/datasets/nirmalsankalana/apple-tree-leaf-disease-dataset

[17] N. R. Billa, B. P. Das, M. Biswal, and M. Okade, "CNN based image resizing forensics for double compressed JPEG images," *J. Inf. Secur. Appl.*, vol. 81, p. 103693, Mar. 2024, doi: 10.1016/j.jisa.2023.103693.

[18] M. T R, M. Gupta, A. T A, V. Kumar V, O. Geman, and D. Kumar V, "An XAI-enhanced efficientNetB0 framework for precision brain tumor detection in MRI imaging," *J. Neurosci. Methods*, vol. 410, p. 110227, Oct. 2024, doi: 10.1016/j.jneumeth.2024.110227.

[19] Bhoomika and G. Verma, "Enhancing Coffee Plant Disease Identification with EfficientNetB0 and Deep Learning," in *2024 8th International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, IEEE, Oct. 2024, pp. 1423–1427. doi: 10.1109/I-SMAC61858.2024.10714676.

[20] M. Niazkar *et al.*, "Applications of XGBoost in water resources engineering: A systematic literature review (Dec 2018–May 2023)," *Environ. Model. Softw.*, vol. 174, p. 105971, Mar. 2024, doi: 10.1016/j.envsoft.2024.105971.

[21] K. M. K. Raghunath, V. V. Kumar, M. Venkatesan, K. K. Singh, T. R. Mahesh, and A. Singh, "XGBoost Regression Classifier (XRC) Model for Cyber Attack Detection and Classification Using Inception V4," *J. Web Eng.*, Apr. 2022, doi: 10.13052/jwe1540-9589.21413.

[22] D. A. A. Pertiwi, K. Ahmad, S. N. Salahudin, A. M. Annegrat, and M. A. Muslim, "Using Genetic Algorithm Feature Selection to Optimize XGBoost Performance in Australian Credit," *J. Soft Comput. Explor.*, vol. 5, no. 1, pp. 92–98, Apr. 2024, doi: 10.52465/joscex.v5i1.302.

[23] M. Heydarian, T. E. Doyle, and R. Samavi, "MLCM: Multi-Label Confusion Matrix," *IEEE Access*, vol. 10, pp. 19083–19095, 2022, doi: 10.1109/ACCESS.2022.3151048.

[24] L.-E. Pomme, R. Bourqui, R. Giot, and D. Auber, "Relative Confusion Matrix: Efficient Comparison of Decision Models," in *2022 26th International Conference Information Visualisation (IV)*, IEEE, Jul. 2022, pp. 98–103. doi: 10.1109/IV56949.2022.00025.