Deep feature representation for automated plant species classification from leaf images

Nikhil Inamdar[1], Manjunath Managuli[1], Uttam Patil[2]
[1]Department of Electronics and Communicatios Engineering, KLS Gogte Institute of Technology Belagavi and Affiliated to Visweravaya Technological University, Belagavi, India
[2]Department of Computer Science and Engineering, Jain College of Engineering, Belagavi and Affiliated to Visweravaya Technological University, Belagavi, India

Article Info

Article history:

Received Aug 23, 2024
Revised Apr 15, 2025
Accepted May 24, 2025

Keywords:

Convolutional neural networks
Plant species classification
Deep learning
Feature representation
Leaf images
Transfer learning

ABSTRACT

Automated plant species classification using leaf images holds immense potential for advancing agricultural research, biodiversity conservation, and ecological monitoring. This study introduces a novel approach leveraging deep feature representation to achieve accurate and efficient classification based on leaf morphology. Convolutional neural networks (CNNs), including VGG16, ResNet50, DenseNet1, Inception, and Xception, are employed to extract high-level features from leaf images, capturing intricate patterns essential for species differentiation. To manage the extensive feature set extracted by these models, optimization techniques such as principal component analysis (PCA), variance thresholding, and recursive feature elimination (RFE) are applied. These methods streamline the feature set, making the classification process more efficient. The optimized features are then trained using classifiers like support vector machine (SVM), k-nearest neighbors (K-NN), decision trees (DT), and naive Bayes (NB), achieving average accuracies of 98.6%, 96.6%, 99.6%, and 99.7%, respectively, across various cross-validation methods. Experimental results on benchmark datasets demonstrate the effectiveness of this approach, achieving state-of-the-art performance in plant species classification. This work underscores the potential of deep feature representation in automated plant species classification, offering valuable insights for applications in agriculture, ecology, and environmental science.

Corresponding Author:

Nikhil Inamdar
Department of Electronics and Communication Engineering, KLS Gogte Institute of Technology Belagavi and Affiliated to Visweravaya Technological University
Jnashodha Campus Udyambhag, Belagavi India
Email: nikhil0870@gmail.com

1. INTRODUCTION

Detecting plant diseases through image analysis is essential in precision agriculture. Traditionally, disease severity has been assessed visually by experts [1]. Although the use of digital cameras and advancements in information technology has increased the adoption of expert systems in agriculture, enhancing plant production [2], these systems still face significant challenges.

Various artificial intelligence (AI) techniques, including K-nearest neighbors (K-NN), logistic regression, decision trees, support vector machines (SVM) [3], and convolutional neural networks (CNN), are extensively used for plant disease detection and classification. These methods often involve image preprocessing to enhance feature extraction. K-NN, a supervised algorithm, classifies by comparing similarities between unlabeled and labeled data [4]. Decision trees, structured like a flowchart, make decisions based on attributes at each node but can face challenges such as overlapping nodes and overfitting [5].

Sharif *et al.* [6] Introduced a technique for detecting diseases in citrus plants using texture features, combining principal component analysis with feature statistics for hybrid feature selection. Patil and Kumar [7] developed a system to detect diseases in soybean leaves by analyzing color, shape, and texture features using a content-based image retrieval approach. Sandika *et al.* [8] proposed a method for identifying grape leaf diseases in complex backgrounds, utilizing features like local binary patterns (LBP) and RGB color statistics, and applying machine learning algorithms such as support vector machines (SVM) and random forest.

The primary research problem addressed in this study is the accurate and efficient classification of plant species using leaf images, overcoming the limitations of traditional methods that rely on manual feature extraction and conventional machine learning techniques. Our unique approach leverages deep feature representation by integrating multiple advanced CNN architectures (VGG16, ResNet50, DenseNet121, Inception, and Xception) to capture intricate patterns essential for species differentiation. Additionally, we employ sophisticated feature optimization techniques such as principal component analysis (PCA), variance thresholding, and recursive feature elimination (RFE) to streamline the feature set, reducing computational complexity and improving efficiency. This combination of deep feature extraction and optimization, along with the use of diverse classifiers (SVM, K-NN, decision trees, naive Bayes), results in state-of-the-art performance with accuracies up to 99.7%.

2. RELATED STUDY

Deep learning architectures have recently shown great effectiveness in tasks like object identification, classification, and segmentation, with CNNs being particularly prominent [9]. For example, Selvaraj *et al.* [10] developed a dataset of banana plant samples from Africa and Southern India, covering 17 diseased classes and one healthy class. Using CNN architectures like ResNet50, InceptionV2, and MobileNetV1, they achieved a 90% accuracy rate. Lu *et al.* [11] explores various classifiers, including SVM, KNN, and Random Forest, for classification. The study [12] underscores the adaptability of CNN-based feature extraction and the potential of different classifiers in accurately classifying agricultural data. Barbedo [13] explores 79 disease detection of 14 different plant species. [14] introduces an optimal feature set for achieving higher classification accuracy in agricultural crop species classification. By combining various features and datasets, the study aims to further optimize classification accuracy and explore different feature combinations to enhance performance. [15] proposes an intelligent system for real-time identification of Indian medicinal herbs based on leaf images, utilizing Raspberry Pi. The developed machine learning models achieve impressive accuracy rates, demonstrating the feasibility of using low-cost hardware for real-world applications in plant identification.

Introducing a CNN based method called D-Leaf for leaf classification, [16] compares different CNN models based on their feature extraction capabilities. The D-Leaf model achieves competitive testing accuracy compared to pre-trained CNN models. Chuanlei *et al.* [17] achieved 97% accuracy in their experiments on wheat plants, using a dataset with six diseased classes and one healthy class. Similarly, Singh *et al.* [18] developed a system for detecting tea leaf diseases with a modified deep convolutional neural network, achieving an average accuracy of 92%. Chakraborty *et al.* [19] conducted experiments on 79 different diseases across 14 plant types using the GoogLeNet architecture, with accuracy scores consistently above 75%. Krizhevsky *et al.* [20] explored various CNN architectures, achieving up to 99% accuracy, with VGG reaching 81% across multiple plant types. Geetharamani and Pandian [3] worked with a dataset containing 38 classes from 14 plant types, attaining a 96% accuracy rate. Traditional machine learning approaches, though effective in plant disease identification, are limited by the sequential nature of image segmentation [21], feature extraction [22], and pattern recognition [23]. While basic CNN models like AlexNet, VGGNet, GoogLeNet, DenseNet, and ResNet have been extensively utilized for plant disease classification, they come with drawbacks such as high parameter demands and slow computation speeds. Despite their strength in capturing both high- and low-level features, these models often struggle with consistently describing local spatial characteristics [24]. [24] Implemented residual learning framework to ease the training mechanism there is 28% relative improvement in COCO object detection dataset.

3. METHODOLOGY

Block schematic of the proposed method is shown in Figure 1. It consists of four stages, namely, pre-processing, deep feature extraction, feature optimization and classification. The research focuses on classification of plant leaf images, many researchers have worked on identification of leaf diseases rather than on identification leaftypes specifically growing in this part of the country and also proved to have rich business value. There exist many varieties of plants; the article proposes 14 plant leaf classification methodology using deep features. Datasets of 14 plants are collected from different research work cited in the literature survey section. Commonly and widely used CNN are used for deep feature extraction, namely, VGG16, Resnet50, InceptionV3, Xception and Densenet121. Models such as VGG16, pre-trained on large datasets like ImageNet, offer a transferable set of features that capture rich visual information from images. ResNet enables effective gradient flow and facilitates the learning of highly abstract features throughout the network layers. InceptionV3 and its variants leverage inception modules, which use multiple convolutions of different kernel sizes within the same layer. Xception, inspired by Inception architecture, replaces traditional convolutions with depth wise separable convolutions. This modification decouples spatial and channel-wise correlations in feature maps, leading to improved feature representation with fewer parameters.

Figure 2 shows the use of VGG16 model for deep feature extraction and its relevant approximate breakdown of the number of features extracted at each layer:

a. Input layer: this layer doesn't produce features directly, but it accepts input images of size (224, 224, 3).
b. Convolutional layers: VGG16 has a total of 13 convolutional layers (including pooling layers).
c. Each convolutional layer typically outputs feature maps of varying sizes, gradually reducing spatial dimensions while increasing depth (number of filters).
d. Fully connected layers: after flattening, VGG16 includes three fully connected layers with decreasing numbers of neurons: 4096, 4096, and 1000 (for ImageNet's 1000 classes).25,088 represent the feature vector extracted from the last convolutional layer before feeding into fully connected layers or classification.Having various CNN models available in the literature, an experiment was conducted to analyze the behavior of different CNN models and the number of features they extract. Table 1 provides detailed performance metrics for these models both with and without feature optimization.

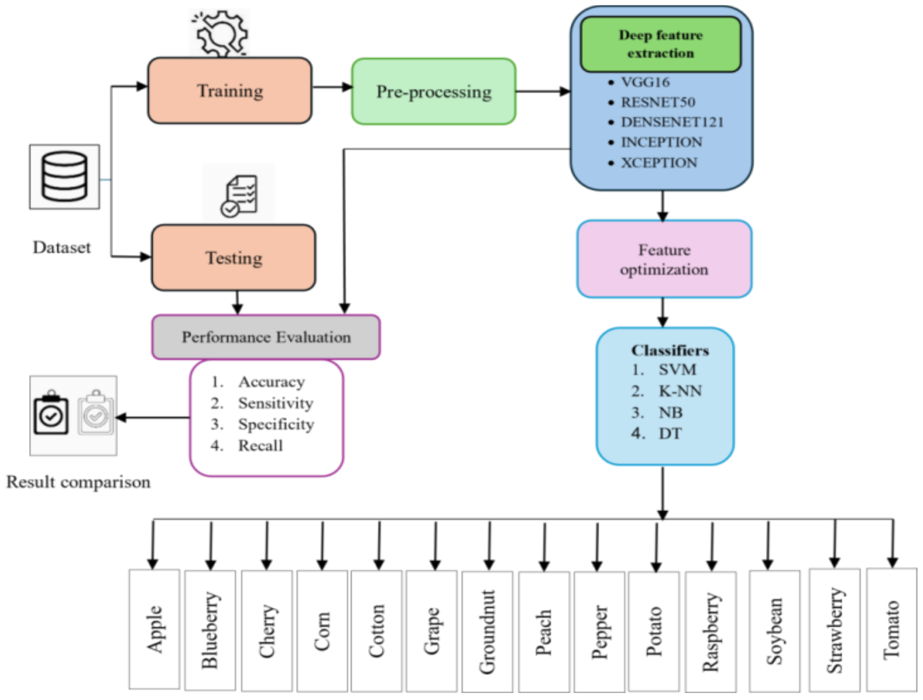


Figure 1. Block schematic of the proposed study

3.1. Principal component analysis (PCA)

It is a method used for dimensionality reduction, widely applied in fields like image processing, data visualization, and machine learning. Mathematically, PCA begins by standardizing the data to ensure uniformity across variables. Then, it computes the covariance matrix of the standardized data:

$$covarience = \frac{1}{(n-1)}(X^T X) \tag{1}$$

Here, X is the $n \times p$ matrix of the standardized data.

PCA then derives the eigenvectors $v_1, v_2, \ldots, v_p$ and their corresponding eigenvalues λ1, λ2, …, λp arranged in descending order. These represent the directions of maximum variance and their magnitudes, respectively.

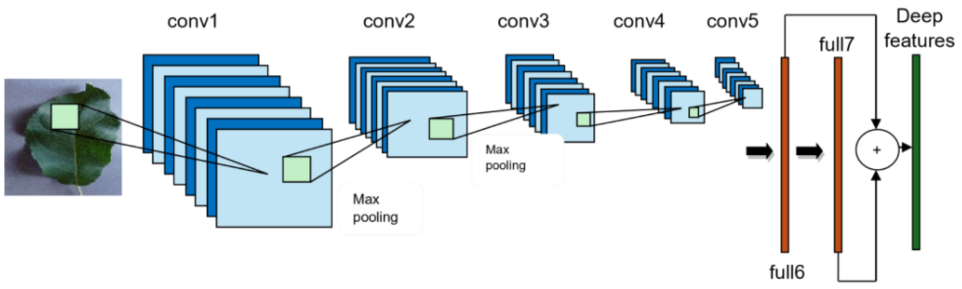


Figure 2. Deep feature extraction methodology

Table 1. Number of features extracted from different CNN models considered

	VGG16	RESNET50	INCEPTIONV3	DENSENET121	XCEPTION
Original number of features	25088	100352	131072	50176	204800
Reduced number of features after PCA	179	185	183	174	172
Accuracy after PCA:	0.7857	0.8928	0.8928	0.875	0.9285
Reduced number of features after variance thresholding	21016	59900	70480	21180	29083
Accuracy after variance thresholding	0.8928	0.9107	0.8214	0.9285	0.9107
Number of features after preliminary PCA	100	100	100	100	100
Reduced number of features after RFE	50	50	50	50	50
Accuracy after RFE:	0.8928	0.8928	0.8571	0.9107	0.9285
Accuracy without optimization:	0.9107	0.9464	0.8392	0.8928	0.9285

3.1.1. Variance thresholding

Variance thresholding is employed as a feature selection technique to enhance the performance of the random forest classifier by eliminating features with low variance that are deemed less informative for classification tasks. The method hinges on the premise that features exhibiting minimal variance across samples are less likely to contribute meaningfully to distinguishing between different classes. For each feature *f* in the dataset, the variance σ_f^2 is computed as the average squared deviation from the mean:

$$\sigma_f^2 = \frac{1}{n}\sum_{i=1}^{n}(x_{i,f} - \bar{x}_f)^2 \tag{2}$$

Here, $x_{i,f}$ represents the value of feature f in sample i, ($\bar{x}_f$) denotes the mean of feature f across all samples, and n signifies the total number of samples.

3.1.2. Recursive feature elimination (RFE)

RFE is a feature selection technique that systematically removes attributes to improve model performance and interpretability. In RFE, a machine learning model is trained on the dataset, and features are recursively pruned based on their importance until the optimal subset is determined. The process begins by fitting a machine learning model (in our case, a random forest classifier) to the dataset with an initial set of features, which may be reduced using principal component analysis (PCA) to manage high-dimensional data effectively:

$$X_{pca} = PCA(X_{train}, n_{components} = 100)$$

Here, X_{train} represents the training data, and X_{pca} denotes the PCA-transformed feature set with reduced dimensions.

3.2. Optimized feature selection

Building upon the mathematical foundation of optimization techniques discussed earlier, this section outlines the specific approach used for feature optimization. The chosen methodology is tailored to enhance the selection of relevant features. Each step is designed to ensure improved model performance and computational efficiency.

3.3. Principal component analysis (PCA)

Principal component analysis (PCA) is employed to reduce the dimensionality of the extracted deep features while preserving 95% of their original variance by setting $n_{components}$ to 0.95. This dimensionality reduction plays a key role in addressing the curse of dimensionality, which can negatively impact model accuracy and efficiency. Additionally, it significantly decreases computational load, enabling faster processing without a notable loss in model implementation. PCA is applied to reduce the dimensionality of the extracted deep features while retaining 95% of their variance ($n_{components}$ =0.95). This reduction helps mitigate the curse of dimensionality and speeds up computation without significantly sacrificing model performance.

After PCA transformation ($X_{train_{pca}}$ and $X_{test_{pca}}$), a random forest classifier ($classifier_{pca}$) is trained and evaluated on the reduced feature set to assess its classification accuracy ($accuracy_{pca}$). Setting PCA ($n_{components}$ =0.55) to retain 55% of variance is a strategic choice that balances dimensionality reduction with the preservation of essential information. By retaining 55% of the variance, the number of features is significantly reduced, which helps alleviate the curse of dimensionality and mitigate over fitting. This reduction simplifies the model and decreases computational complexity, while still maintaining enough variance to ensure that crucial information is preserved for accurate predictions. This approach effectively addresses the trade-off between reducing feature space and retaining significant data characteristics, thus optimizing the performance of the model. Also, Table 2 gives the performance of the model for different percentage of variance.

3.3.1. Variance thresholding

Variance thresholding ($VarianceThreshold$) is used to remove features with low variance. The threshold is set dynamically based on the variance of each feature ($threshold = (0.8 * (1 - 0.8))$). This technique is beneficial for eliminating features that do not vary much within the dataset, potentially reducing noise and improving model robustness.

The random forest classifier ($classifier\_var$) is trained and evaluated on the selected features (X_train_var and X_test_var), and its accuracy ($accuracy\_var$) is computed to compare with PCA. The use of $VarianceThreshold(threshold = (4 * (1 - 0.8)))$ with a threshold of 0.8 (or 4×0.2) is designed to filter out features with low variance, which are less likely to contribute meaningful information to the model. The threshold value of 0.8 is chosen to remove features that have very low variance—specifically, those with variance less than 20% of the overall variance. This approach enhances model efficiency by focusing on more informative features while discarding those that contribute little to predictive performance. Table 3 provides the variation of the model performance for different variance threshold values.

Table 2. Performance of the model for different variance during optimization

Variance	0.95	0.75	0.55	0.45	0.35
Original number of features: 25088	25088	25088	25088	25088	25088
Reduced number of features after PCA: 90	179	90	43	27	15
Accuracy after PCA: 0.8214285714285714	0.78	0.8214	0.85	0.82	0.8
Reduced number of features after Variance Thresholding: 21016	21016	21016	21016	21016	21016
Accuracy after Variance Thresholding: 0.8928571428571429	0.89	0.8928	0.89	0.892	0.892
Number of features after preliminary PCA: 100	100	100	100	100	100
Reduced number of features after RFE: 50	50	50	50	50	50
Accuracy after RFE: 0.8214285714285714	0.85	0.8214	0.8214	0.8214	0.8214

Table 3. Performance of the model for different threshold during optimization

Threshold	0.8	0.9	1	1.2	2	4	8	16
Number of images	280	280	280	280	280	280	280	280
Original number of features	25088	25088	25088	25088	25088	25088	25088	25088
Reduced number of features after PCA	43	43	43	43	43	43	43	43
Accuracy after PCA	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85
Redatures after Variance Thresholding	21016	20863	20734	20490	19688	18102	15959	13141
Accuracy after Variance Thresholding	0.89	0.89	0.83	0.875	0.91	0.91	0.875	0.857
Number of features after preliminary PCA	200	200	200	200	200	200	200	200
Reduced number of features after RFE	100	100	100	100	100	100	100	100
Accuracy after RFE	0.85	0.85	0.85	0.85	0.85	0.85	0.85	0.85
Accuracy without optimization	0.91	0.91	0.91	0.91	0.91	0.91	0.91	0.91

3.3.2. Recursive feature elimination (RFE)

RFE is applied after an initial PCA reduction (PCA ($n\_components = 100$)) to further select the most informative features ($n\_features\_to\_select = 50$). RFE iteratively removes less important features based on their contribution to model accuracy, using a Random Forest estimator ($rfe\_estimator$). In this approach, a random forest classifier is employed to rank feature importance due to its capacity to manage large feature sets and generate reliable importance scores. The RFE process is configured to select the top 100 features, systematically removing 5 features at each iteration. Random forest classifiers are used to rank feature importance, leveraging its ability to handle large feature sets and provide robust importance scores. The RFE configuration aims to select 100 features, removing 5 features in at each iteration. This approach involves training the model on the full feature set, ranking features based on their importance, and then eliminating the least significant ones. This process is repeated until the desired number of features is reached, ensuring that only the most impactful features are retained for enhanced model efficiency and performance. Table 4 provides the variation of the model performance for different estimator to reduced features values.

Table 4. Performance of the model for different ratios of features to iteration

Reduced features	150/100	200/100	100/100
Original number of features: 25088	25088	25088	25088
Reduced number of features after PCA: 43	43	43	43
Accuracy after PCA: 0.8571428571428571	0.85	0.85	0.85
Reduced number of features after Variance Thresholding: 21016	21016	21016	21016
Accuracy after Variance Thresholding: 0.8928571428571429	0.89	0.89	0.89
Number of features after preliminary PCA: 150	150	200	100
Reduced number of features after RFE: 100	100	100	100
Accuracy after RFE: 0.8571428571428571	0.82	0.85	0.83

3.3.3. Classification

This study evaluates four classifiers: random forest, K-NN, SVM, and naïve Bayes. Random forest, an ensemble method, constructs multiple decision trees, offering robustness and high accuracy with complex datasets. K-NN classifies data based on the nearest neighbors; it is simple but can be computationally intensive with large datasets. SVM is a powerful supervised algorithm that finds the optimal hyperplane for class separation in high-dimensional spaces but requires careful tuning. Naïve Bayes, a probabilistic classifier based on Bayes' theorem, assumes feature independence and is effective for specific tasks like text classification. These classifiers were selected for their unique approaches and strengths, offering a comprehensive comparison of their performance. The performance of these models is depicted in Figure 3, with corresponding classification metrics for different CNN models provided in Table 5.

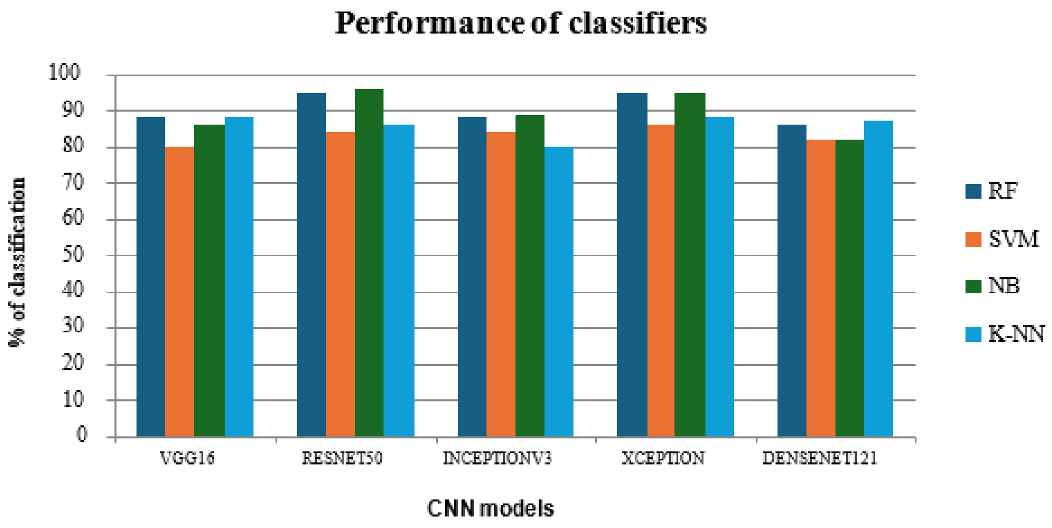


Figure 3. Performance of classifiers for different CNN models

Table 5. Classification metrics for different CNN models

CNN	Classifier	Accuracy	Precision	Recall	F1-Score
VGG16	RF	95	97	95	95
	SVM	91	95	91	91
	NB	95	96	95	95
	K-NN	91	92	91	91
RESNET50	RF	95	96	95	95
	SVM	84	91	84	84
	NB	96	97	96	97
	K-NN	86	90	86	85
INCEPTIONV3	RF	88	90	88	88
	SVM	84	82	84	85
	NB	89	90	89	89
	K-NN	88	91	88	87
XCEPTION	RF	95	97	95	95
	SVM	86	92	86	87
	NB	95	96	95	95
	K-NN	88	91	88	87
DENSENET121	RF	95	97	95	95
	SVM	91	95	91	91
	NB	95	96	95	95
	K-NN	91	92	91	91

4. RESULTS AND DISCUSSION

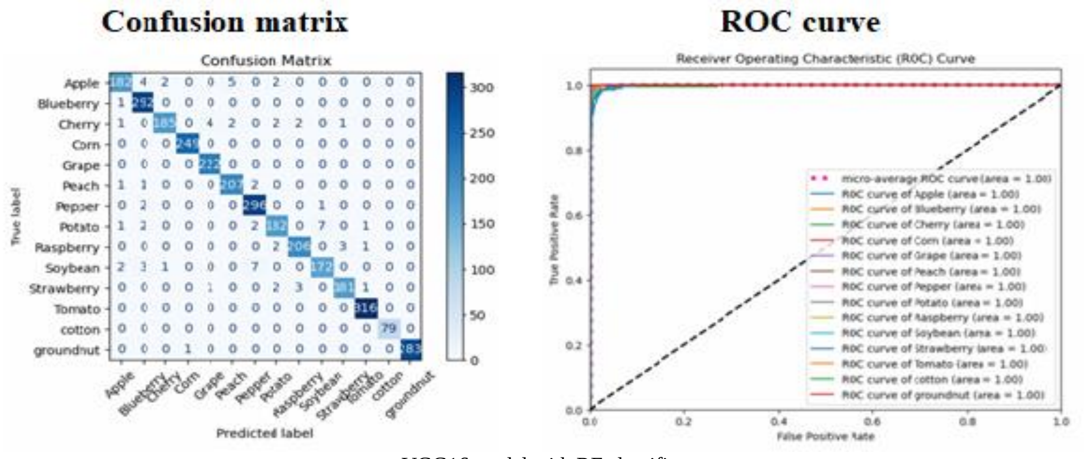
The performance evaluation of various CNN models, including VGG16, ResNet50, InceptionV3, DenseNet121, and Xception, before and after applying feature optimization techniques, provides valuable insights into the effectiveness of these models and the impact of optimization on classification accuracy. Initially, models like ResNet50 and Xception performed well even without optimization, indicating their inherent capability to capture and represent intricate patterns in leaf images. However, feature optimization techniques such as PCA, variance thresholding, and RFE significantly enhanced the models' performance by reducing dimensionality, filtering out less informative features, and systematically removing less important features. This led to improved computational efficiency and accuracy, with optimized feature sets providing a more efficient foundation for future model development. Table 6 provides the details of the hyper parameters used to set the classifiers. The comparative performance of the models highlights the importance of feature optimization in achieving state-of-the-art results. Although some models achieved high accuracy without optimization, the application of PCA, Variance Thresholding, and RFE provided additional benefits in terms of efficiency and robustness. Figure 4 shows the confusion matrix and ROC curve for various CNN models with selected classifiers, demonstrating strong classification accuracy as detailed in Table 5. The confusion matrix and ROC curve analyses further illustrate the models' effectiveness in distinguishing between different plant species, with higher AUC values reflecting better class discrimination. This in-depth analysis underscores the significance of integrating advanced CNN architectures with feature optimization techniques, making the proposed model a valuable tool for precision agriculture, biodiversity conservation, and ecological monitoring.

4.1. Comparison with existing methods

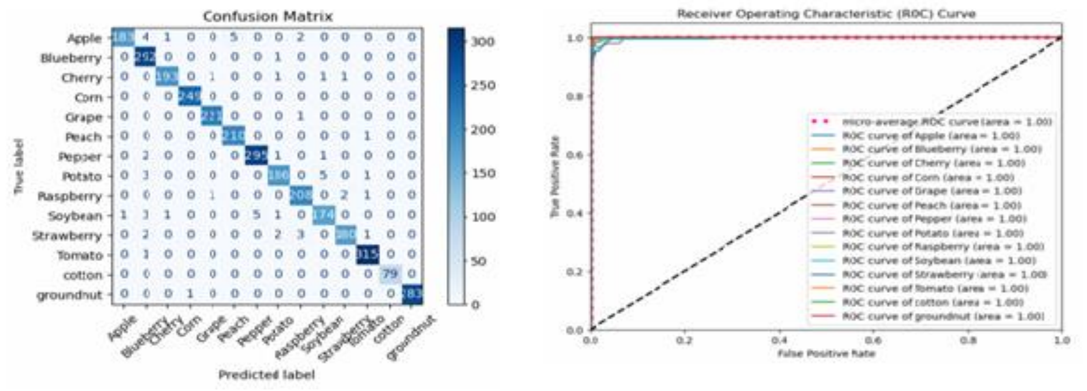
The dataset under consideration consists of 20,357 images representing fourteen classes of plant leaves. A comparative analysis, highlights the performance differences between handcrafted and deep features. While deep features exhibit superior performance, they also present challenges, such as the need for substantial datasets and significant computational resources. Furthermore, Table 6 illustrates that this work surpasses conventional research methodologies, particularly in handling a larger number of classes.

Table 6. Comparison with related work

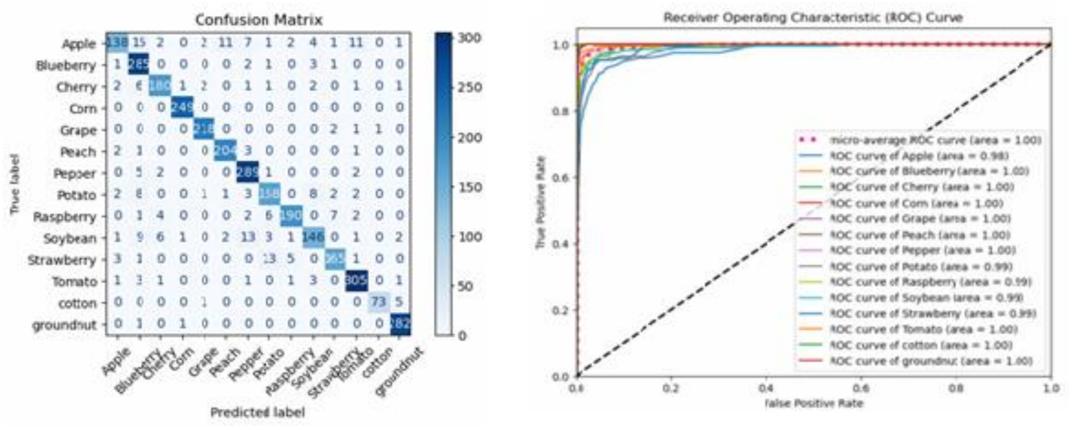
Sl no	Reference no	Method	No of classes	Accuracy
1	[25]	VGG16	4	90.40
2	[20]	M-SVM	4	97.20
3	[21]	DWT, COLOR HISTOGRAM	3	98.63
4	[22]	SHUFFLENETV1	4	97.79
5	[23]	DEEP FEATURE + LBP	4	98.80
6	Proposed	Deep features	14	99.7



VGG16 model with RF classifier



DenseNet model with NB classifier



InceptionV3 model with NB classifier

Figure 4. Confusion matrix and ROC curve of CNN models

5. CONCLUSION

The study demonstrates the efficacy of deep feature representation in the automated classification of plant species using leaf images. By employing CNN models such as VGG16, ResNet50, DenseNet121, Inception, and Xception, we successfully captured high-level, discriminative features essential for accurate species differentiation. The application of optimization techniques like PCA, Variance Thresholding, and RFE further enhanced the efficiency of the feature set, leading to high classification accuracies when combined with classifiers such as SVM, K-NN, DT, and NB. The achieved results, with accuracies reaching up to 99.7%, underscore the potential of this approach in advancing agricultural research, biodiversity conservation, and ecological monitoring. Future work will focus on expanding the dataset and exploring additional optimization strategies to further refine the classification process.

FUNDING INFORMATION

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

AUTHOR CONTRIBUTIONS STATEMENT

Nikhil Inamdar conceived the study, designed the methodology, and performed the experiments. Manjunath Manguli contributed to data analysis, interpretation, and technical validation. Uttam Patil assisted with manuscript drafting, critical revisions, and final approval of the version to be published. All authors have read and agreed to the published version of the manuscript.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Nikhil Inamdar	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	
Manjunath Managuli										✓		✓		
Uttam Patil										✓		✓		

C	: **C**onceptualization	I	: **I**nvestigation	Vi	: **Vi**sualization
M	: **M**ethodology	R	: **R**esources	Su	: **Su**pervision
So	: **So**ftware	D	: **D**ata Curation	P	: **P**roject administration
Va	: **Va**lidation	O	: Writing - **O**riginal Draft	Fu	: **Fu**nding acquisition
Fo	: **Fo**rmal analysis	E	: Writing - Review & **E**diting		

CONFLICT OF INTEREST STATEMENT

Authors declare that there is no conflict of interest

DATA AVAILABILITY

The data used in this study are publicly available and can be accessed from open sources as cited within the manuscript or can be contacted to author

REFERENCES

[1] A. A. Bharate and M. S. Shirdhonkar, "A review on plant disease detection using image processing," in *2017 International Conference on Intelligent Sustainable Systems (ICISS)*, Dec. 2017, pp. 103–109. doi: 10.1109/iss1.2017.8389326.

[2] P. Zhao, G. Liu, M. Li, and D. Li, "Management information system for apple diseases and insect pests based on GIS," *Nongye Gongcheng Xuebao/Transactions of the Chinese Society of Agricultural Engineering*, vol. 22, no. 12, pp. 150–154, 2006, doi: 10.3969/j.issn.1002-6819.2006.12.029.

[3] G. Geetharamani and J. A. Pandian, "Identification of plant leaf diseases using a nine-layer deep convolutional neural network," *Computers & Electrical Engineering*, vol. 76, pp. 323–338, Jun. 2019, doi: 10.1016/j.compeleceng.2019.04.011.

[4] E. M. F. El Houby, "A survey on applying machine learning techniques for management of diseases," *Journal of Applied Biomedicine*, vol. 16, no. 3, pp. 165–174, Aug. 2018, doi: 10.1016/j.jab.2018.01.002.

[5] C.-C. Yang *et al.*, "Application of decision tree technology for image classification using remote sensing data," *Agricultural Systems*, vol. 76, no. 3, pp. 1101–1117, Jun. 2003, doi: 10.1016/s0308-521x(02)00051-3.

[6] M. Sharif, M. A. Khan, Z. Iqbal, M. F. Azam, M. I. U. Lali, and M. Y. Javed, "Detection and classification of citrus diseases in agriculture based on optimized weighted segmentation and feature selection," *Computers and Electronics in Agriculture*, vol. 150, pp. 220–234, Jul. 2018, doi: 10.1016/j.compag.2018.04.023.

[7] J. K. Patil and R. Kumar, "Analysis of content based image retrieval for plant leaf diseases using color, shape and texture features," *Engineering in Agriculture, Environment and Food*, vol. 10, no. 2, pp. 69–78, Apr. 2017, doi: 10.1016/j.eaef.2016.11.004.

[8] B. Sandika, S. Avil, S. Sanat, and P. Srinivasu, "Random forest based classification of diseases in grapes from images captured in uncontrolled environments," in *2016 IEEE 13th International Conference on Signal Processing (ICSP)*, Nov. 2016, pp. 1775–1780. doi: 10.1109/icsp.2016.7878133.

[9] S. Uğuz and N. Uysal, "Classification of olive leaf diseases using deep convolutional neural networks," *Neural Computing and Applications*, vol. 33, no. 9, pp. 4133–4149, Aug. 2020, doi: 10.1007/s00521-020-05235-5.

[10] M. G. Selvaraj *et al.*, "AI-powered banana diseases and pest detection," *Plant Methods*, vol. 15, no. 1, Aug. 2019, doi: 10.1186/s13007-019-0475-z.

[11] J. Lu, J. Hu, G. Zhao, F. Mei, and C. Zhang, "An in-field automatic wheat disease diagnosis system," *Computers and Electronics in Agriculture*, vol. 142, pp. 369–379, Nov. 2017, doi: 10.1016/j.compag.2017.09.012.

[12] G. Hu, X. Yang, Y. Zhang, and M. Wan, "Identification of tea leaf diseases by using an improved deep convolutional neural network," *Sustainable Computing: Informatics and Systems*, vol. 24, p. 100353, Dec. 2019, doi: 10.1016/j.suscom.2019.100353.

[13] J. G. Arnal Barbedo, "Plant disease identification from individual lesions and spots using deep learning," *Biosystems Engineering*, vol. 180, pp. 96–107, Apr. 2019, doi: 10.1016/j.biosystemseng.2019.02.002.

[14] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, "A comparative study of fine-tuning deep learning models for plant disease identification," *Computers and Electronics in Agriculture*, vol. 161, pp. 272–279, Jun. 2019, doi: 10.1016/j.compag.2018.03.032.

[15] K. P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," *Computers and Electronics in Agriculture*,

vol. 145, pp. 311–318, Feb. 2018, doi: 10.1016/j.compag.2018.01.009.
[16] B. Liu, Y. Zhang, D. He, and Y. Li, “Identification of Apple leaf diseases based on deep convolutional neural networks,” *Symmetry*, vol. 10, no. 1, p. 11, Dec. 2017, doi: 10.3390/sym10010011.
[17] Z. Chuanlei, Z. Shanwen, Y. Jucheng, S. Yancui, and C. Jia, “Apple leaf disease identification using genetic algorithm and correlation-based feature selection method,” *International Journal of Agricultural and Biological Engineering*, vol. 10, no. 2, pp. 74–83, 2017, doi: 10.3965/j.ijabe.20171002.2166.
[18] S. Singh, S. Gupta, A. Tanta, and R. Gupta, “Extraction of multiple diseases in apple leaf using machine learning,” *International Journal of Image and Graphics*, vol. 22, no. 03, Feb. 2021, doi: 10.1142/s021946782140009x.
[19] 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)*, Jan. 2021, pp. 147–151. doi: 10.1109/icrest51555.2021.9331132.
[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
[21] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv:1409.1556*, Sep. 2014.
[22] C. Szegedy *et al.*, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015, pp. 1–9. doi: 10.1109/cvpr.2015.7298594.
[23] F. N. Iandola, M. W. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, “DenseNet: Implementing efficient convnet descriptor pyramids,” *arXiv preprint arXiv:1404.1869*, 2014.
[24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.
[25] L. Cai, J. Zhu, H. Zeng, J. Chen, C. Cai, and K.-K. Ma, “HOG-assisted deep feature learning for pedestrian gender recognition,” *Journal of the Franklin Institute*, vol. 355, no. 4, pp. 1991–2008, Mar. 2018, doi: 10.1016/j.jfranklin.2017.09.003.

BIOGRAPHIES OF AUTHORS

Nikhil Inamdar assistant professor at KLS Gogte Institute of Technology, which is affiliated with Visvesvaraya Technological University, located in Belagavi, India. Holds a Master's degree (M.Tech.) in Industrial Electronics from Visvesvaraya University, Belagavi, showcasing his expertise in the field. Currently pursuing a Ph.D., demonstrating his commitment to furthering his knowledge and contributing to academia. In his academic journey, he has a keen interest in areas such as artificial intelligence (AI), machine learning (ML), deep learning (DL), and embedded systems. These areas of interest reflect passion for cutting-edge technologies and their application in various fields. As an educator and researcher, strives to bridge the gap between theory and practice, aiming to make significant contributions to the advancement of these fields. He can be contacted at email: nikhil0870@gmail.com and njinamdar@git.edu.

Manjunath Managuli is an accomplished associate professor with a decade of dedicated service in academia. Currently affiliated with the esteemed KLS Gogte Institute of Technology, Dr. Managuli has established himself as a dynamic educator and a committed researcher. With a passion for teaching, Dr. Managuli has been actively involved in shaping the academic journey of students, imparting knowledge in [mention the specific field or subject]. His engaging teaching methods and ability to connect with students have garnered admiration and respect within the academic community. He can be contacted at email:manjunathm@git.edu.

Uttam Patil is an accomplished professor and HOD Computer Science Department with a more than decade of dedicated service in academia. Currently affiliated with the esteemed Jain College of Engineering, Dr. Uttam Patil has established himself as a dynamic educator and a committed researcher. With a passion for teaching, Dr. Uttam Patil has been actively involved in shaping the academic journey of students, imparting knowledge. His engaging teaching methods and ability to connect with students have garnered admiration and respect within the academic community He can be contacted at email: uttampatil@jainbgm.in.