International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 8, ISSN No. 2455-2143, Pages 166-173
Published Online December 2022 in IJEAST (http://www.ijeast.com)

A CONVOLUTIONAL NEURAL NETWORK BASED SYSTEM FOR PLANT LEAF RECOGNITION IN SUB-SAHARAN AFRICA

E.B Amadi, C.K Ezemandu, K. Nwachukwu, I.J Ezika, D.O Oyeka
Department of Electronic and Computer Engineering,
University of Nigeria, Nsukka

Abstract: **Rapid plant leaf recognition is beneficial in the field of ecology and botany. This paper offers a fully automated technique of plant recognition using a Convolutional neural network (CNN).This project was deployed on a web-based system that allows users to take pictures of leaves and upload them to a server. MobileNet Architecture was used for easy implementation on the web. The server performs preprocessing and feature extraction on the image taken to create a pattern and then compares it with the images in the database to predict its category. 18 different plant species that are indigenous to Sub-Saharan Africa, a total of 18,000 photos were collected. Following that, morphological features of the leaf are extracted and sent into a machine learning algorithm. In the extraction process, a convolutional neural network is deployed. 90% of the images collected are used to train the model. Finally, the proposed methodology's recognition rate and accuracy are assessed. The built model attained a high accuracy of 97.41%**

Key words: **Convolutional Neural Network (CNN), Artificial Intelligence, Machine Learning, Computer Vision**

I. INTRODUCTION

Speedy identification of plant species is important for the work of a botanist. This is especially important when the plant specie is new, or one they are not familiar with. However, the most common methods to identify plants correctly and easily are manual-based, depending on the morphological characteristics. Thus, many of the processes involved in classifying these plant species are dependent on knowledge accumulation and skills of human beings. This process of manual recognition is often laborious and time-consuming. The problem is further compounded because some plant leaves look very similar to each other. It is undesirable to spend more time than necessary identifying new species and this has led many researchers to conduct studies to support the automatic classification of plants based on their physical characteristics.

There are a variety of classification methods in plant taxonomy, but some of them are difficult to operate, with poor practicality. For effective classification, these methods require the recognition system to have good taxonomic technology. These methods also necessitate the manual acquisition of large data which affects efficiency. Plant leaf recognition will be much useful in plant disease research, protection, agriculture, and forestry applications. The automatic identification of species is possible by analyzing leaf images. A single technology is not enough due to the complexity of leaf shape and pattern. Plant classification mainly uses leaf shape, leaf colour, leaf venation, leaf texture and morphological features.

Some existing systems are linked to this project, such as LeafSnap, Plantnet, and PlantSnap. The goal of this project is to close the gap between international and indigenous plant species in Western Africa. This study focuses on identifying local plant species in Sub-Saharan Africa. The system is deployed to an embedded device, such as a mobile phone, to make it accessible to the general public and researchers. For plant recognition, the web application will be made more user-friendly.

The purpose of the project is to develop an image recognition model that will be used in a web application to solve locally available plant species identification and recognition difficulties that our botanists, experts, and individuals face reducing plant identification sample error.

1.1Related Works

From a machine learning perspective, plant identification is a supervised classification problem [1]. The development of feature detection, extraction, and encoding techniques for computing characteristic feature vectors was the main emphasis of research on automated species identification over the past ten years. It used to be a problem-specific process to create and coordinate such procedures, resulting in a model tailored to the unique applications. Model-free approaches aim to overcome the described limitations of model-based approaches. They do not employ application-specific knowledge and therefore promise a higher degree of generalization across different classes, i.e species and their organs. The core concept of model-free approaches is the

detection of characteristics interest points and their description using generic algorithms, such as scale-invariant feature transform (SIFT), speeded-up robust features (SURF) and histogram of gradients (HOG). These descriptors capture visual information in a patch around each interest point as orientation of gradients and have been successfully used for manifold plant classification studies[2]. That work comparatively evaluate alternative parts of a model-free image classification pipeline for plant species identification. They found the SURF detector in combination with the SIFT local shape descriptor to be superior to other detector descriptor combinations. For encoding interest points, in order to form a characteristics image descriptor for classification, they found the Fisher Kernel encoding to be superior.

[3]Employed a probabilistic neural network (PNN) for classifying feature vectors and Principal Component Analysis (PCA) for reducing the dimensions of the input vector to be fed to the PNN. Implement on a general-purpose automated leaf recognition for the plant classification, 12 leaf features are extracted and orthogonal zed into 5 principal variables which include leaf area, leaf diameter, physiological length, leaf perimeter, and physiological width. The model was trained by 1800 leaves to classify 32 kinds of plants with an accuracy greater than 90% with the Flavia dataset.

The researchers in[4]worked on a model CNN-LSTM architecture, where the CNN layer is embedded with long short-term (LSTM). Their result shows that CNN-LSTM outperforms the convectional CNN because of the use of LSTM the layer which captures significant features lost during training. Several state-of-the-art model architectures were trained, with the proposed model attaining a performance of 95.06%.

Ahmad et al [5]proposed a different approach that involves, the identification of plant using extraction of leaf gas by analyzing the ratio of chlorophyll (Ch) and Nitrogen(N) in both dead and healthy leaves, alongside other features like colour, texture, and geometric features using CNN implemented on MATLAB. Two datasets were used to achieve this work 70% Caltech and 30% Flavia dataset, the model attained an accuracy of 98%.

II. IMAGE DATA ACQUISITION AND PROCESSING

The leaf images are acquired through the following means: Cameras, public datasets like Kaggle [6], TensorFlow datasets [7], Plant Phenotyping datasets[8]. Other sources include web scraping, data augmentation with the Keras preprocessing module[9] and the Augmentor python package[10]. The first two sources are applied for specific classes, while the last sources are used for all the classes.

The data used for training the model were in jpeg image format. They had varied pixel resolutions with each image having three channels (RGB). We had 18 folders containing 18 distinct classes of plant leaves. The classes are; apple, bitter leaf, blueberry, cassava, cherry, curry leaf, grape, guava, mango, oha, peach, potato, raspberry, soybean, strawberry, tomato, ugwu and waterleaf. Out of the total classes listed the following classes were obtained locally from our own image collection process, they are bitter leaf, cassava, curry leaf, guava, mango, oha, ugwu and waterleaf. The remaining classes were obtained from an online plant leaf dataset at https:// data.mendeley. com/datasets/ tywbtsjrjv/ 1.Each of the classes used had exactly 1000 images per class, this resulted in a total of 18,000 images gathered. The image data was split into training, validation and test data in the ratio of 0.9:0.09:0.1 respectively. The total images in the training folder were 16,200 images with 900 images in each class. The total validation images were 1,620 with 90 images in each class. 180 images were then set aside for the final testing of the trained model with Python.

The collected images are split into train, test, and validation datasets. The training dataset is used to train the model, while the test and validation datasets are used to test and evaluate the model.

The images are preprocessed with Keras. preprocessing module, and NumPy. The images are loaded and converted to a 224 x 224 PIL format before it is loaded into a NumPy array. Using NumPy, the dimensions of the array are expanded. The data is then normalized to the range [-1,1] using the Keras. preprocessing module. This works by dividing each of the pixel values by 255. After the above process, the data is ready for training.

A lightweight, fast, accurate, and computationally inexpensive model is required for the proposed framework. For this, the mobileNet v1 model architecture was chosen for the project.

2.1 Training from Pre-Trained Models

To train a model from scratch, you would need high computation power and a high training time, with a large amount of data. Using transfer learning, we utilize available resources and data by taking the weights of a pre-trained model, freezing them, and adding a fully connected layer to be trained with our data. This results in a more accurate model in less training time. The pre-trained model, MobileNet with ImageNet Large Scale Visual Recognition Challenge (ILSVRC) dataset[11], [12], consists of 1.2 million training images with 1000 classes. The fully connected layer for the pre-trained model is removed and replaced with a new one. During training, the pre-trained model extracts meaningful features from the data. It does this by propagating the input images forward and taking the layers' outputs as our features. The obtained feature maps are then used to update the weights, and train the fully connected layer for our specific task.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 8, ISSN No. 2455-2143, Pages 166-173
Published Online December 2022 in IJEAST (http://www.ijeast.com)

2.2 Model Architecture

The initial weights from the model were loaded from ImageNet. ImageNet is a collection of about 15 million tagged high-resolution photos organized into over 22,000 categories. ImageNet contains about 1000 distinct images in each class for 1000 different classes, this includes 1.2 million training images, 50,000 validation images and 150,000 testing images. The training process optimizes the loss by updating these weights. Transfer learning saves training time because the initial weights are not completely random from the onset, rather they have already been optimized for a specific dataset and can be modified to fit our dataset. The input pixel resolution was specified for the model chosen, the image row and column size were set to 224 pixels. The initialized activation function for the model was the softmax activation function. This model has 86 layers. The first layer is the input layer where the input image is resized with background preprocessing techniques available in Keras to ensure that the input image size is a 224 by 224-pixel image with up to 3 channels representing Red, Green and Blue (RGB).

III. MODEL TRAINING

The dataset was made to be balanced to prevent any class from performing poorly due to a smaller number of images in that class. Each class had exactly 900 training images to ensure they have equal influence on the outcome of the model. The learning rate of the model was set to 0.00001. This learning rate was small enough for the model to slowly achieve gradient descent and reduce the loss.

Early stopping was used to stop the training process when the loss cannot undergo further reduction. For each epoch, the model checkpoint was saved as far as the validation loss of the model improved from the initial starting value. To obtain the best model in the training process, when the loss stops reducing the training stops after 3 epochs to save computing power.

The model was trained for 50 epochs with a batch size of 32. This resulted in 506 steps per epoch and an average per step training time of 1 second per step. The accuracy obtained from the first epoch of training was 60.71%. This is an advantage of transfer learning. The model would have taken a long time to achieve high accuracy when training from scratch. The training process continued for only 26 epochs before the early stopping function halted the training. The final training metrics are shown in Table 1.

Table 1: Training Results of the Model Used

Loss	Accuracy	Validation Loss	Validation Accuracy
0.0796	97.41%	0.0625	98.37%

The process of training the model is to minimize the loss and improve accuracy. From the loss graph below the loss dropped from 1.8518 to 0.0796. The process of reducing loss during training is known as gradient descent. Gradient Descent tries to find the global minima of the loss and then calls the early stopping function when the loss no longer appreciably reduces. After, this the model is saved in a .h5 format. A graph of the optimization loss is shown in Fig. 1 while a graph of accuracy is shown in Fig. 2.

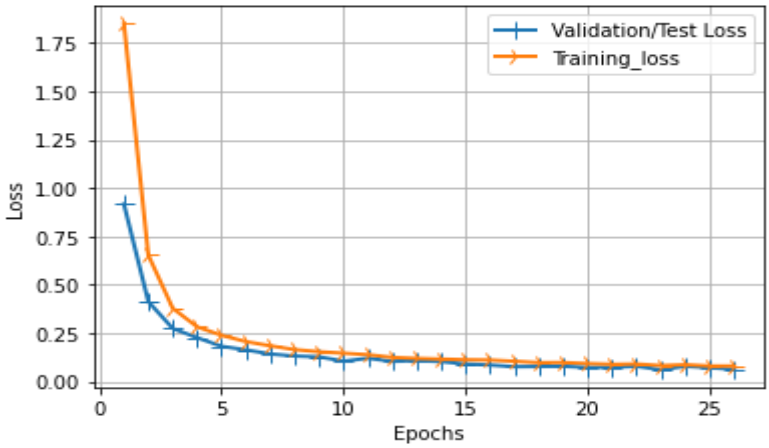

Fig. 1. Graph of the Loss Optimization during training for 26 Epochs

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 8, ISSN No. 2455-2143, Pages 166-173
Published Online December 2022 in IJEAST (http://www.ijeast.com)

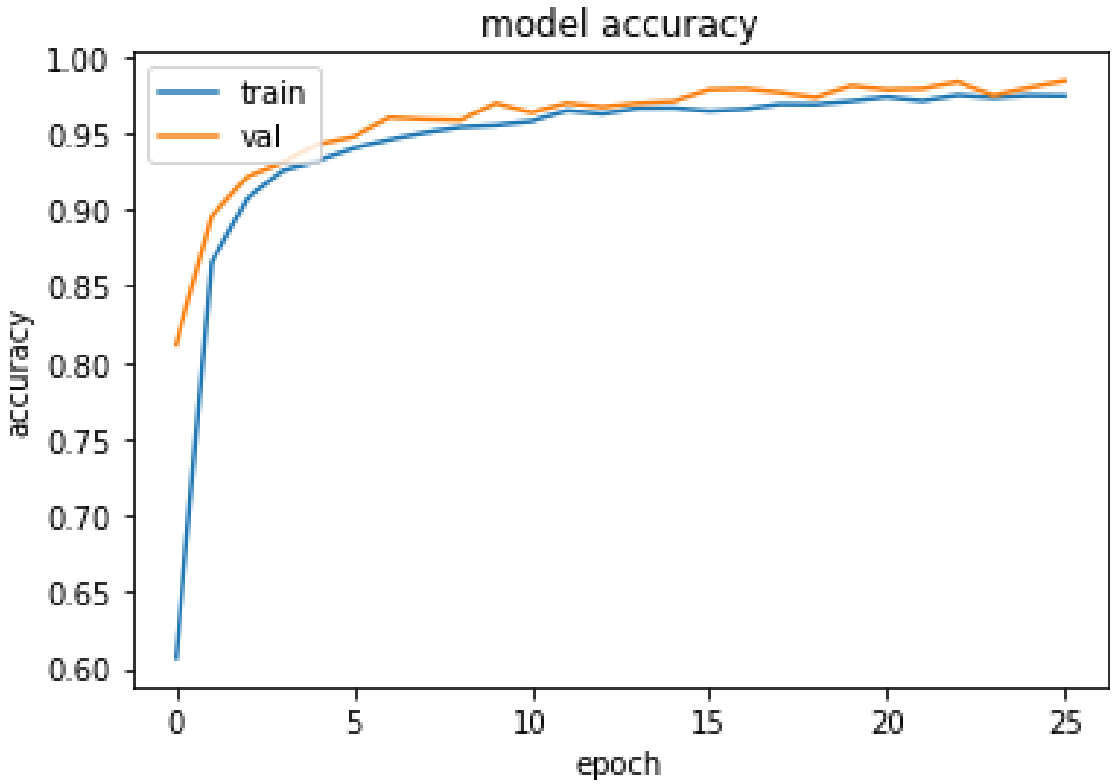

Fig. 2. Graph of Improving Accuracy for 26 Epochs

IV. MODEL PERFORMANCE

4.1 Recall, Precision and F1 Score

The precision, recall and f1 score values gives more insights into the performance of the model beyond the accuracy of the model. It helps us know how well the model performs for each class. A brief description of each of these evaluation metrics would be shown. These metrics where obtained using the classification module within the scikit-learn library.

Recall: This shows how many of the positive classes we got correctly. It is gotten from the formula below.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Precision: This shows how many predictions the classifier got right out of all the predictions

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

F1 Score: This is a metric that attempts to measure both recall and precision.

$$\text{F1} - \text{score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Precision} + \text{Recall}}$$

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 8, ISSN No. 2455-2143, Pages 166-173
Published Online December 2022 in IJEAST (http://www.ijeast.com)

Table 2: Precision, Recall, F1 Score and Support Performance of the Trained Model

S/N	Precision	Recall	F1-score	Support
0	1.00	0.98	0.99	90
1	0.90	0.98	0.94	90
2	1.00	0.98	0.99	90
3	1.00	1.00	1.00	90
4	0.99	1.00	0.99	90
5	0.99	0.98	0.98	90
6	1.00	1.00	1.00	90
7	0.96	0.90	0.93	90
8	0.97	0.99	0.98	90
9	0.93	0.97	0.95	90
10	0.99	1.00	0.99	90
11	0.98	0.96	0.97	90
12	1.00	1.00	1.00	90
13	0.95	1.00	0.97	90
14	1.00	0.99	0.99	90
15	0.99	0.98	0.98	90
16	0.97	0.93	0.95	90
17	0.97	0.93	0.95	90
Accuracy			0.98	1620

4.2 Confusion Matrix

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. A confusion matrix is a visual representation of the performance of the model on the validation dataset. The matrix compares the actual target values with those predicted by the machine learning model. This gives a holistic view of how well one's classification model is performing and what kind of errors it is making. Each column represents each of the classes in the model. The classes are arranged alphabetically and indexed from 0-17.

Table 2: Index Identifying each Class on the Confusion Matrix

Apple: 0	Bitter leaf: 1	Blueberry: 2	Cassava: 3	Cherry: 4	Curry leaf: 5
Grape: 6	Guava: 7	Mango: 8	Oha: 9	Peach: 10	Potato: 11
Raspberry: 12	Soybean: 13	Strawberry: 14	Tomato: 15	Ugwu: 16	Waterleaf: 17

From the data obtained from the confusion matrix, Guava has the least number of images correctly classified. Some of the Guava images were misclassified as Bitter leaf(1), Mango(8), Oha(9), Ugwu(16) and Waterleaf(17). Other classes with a relatively low accuracy are Ugwu and Waterleaf.

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 8, ISSN No. 2455-2143, Pages 166-173
Published Online December 2022 in IJEAST (http://www.ijeast.com)

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	88	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
1	0	88	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
2	0	0	88	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
3	0	0	0	90	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	90	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	1	0	0	0	88	0	1	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	90	0	0	0	0	0	0	0	0	0	0	0
7	0	1	0	0	0	0	0	81	2	2	0	0	0	0	0	0	2	2
8	0	1	0	0	0	0	0	0	89	0	0	0	0	0	0	0	0	0
9	0	3	0	0	0	0	0	0	0	87	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	90	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	86	0	4	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	90	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	90	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	89	1	0	0
15	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	88	0	0
16	0	2	0	0	1	1	0	0	0	2	0	0	0	0	0	0	84	0
17	0	2	0	0	0	0	0	1	0	2	0	0	0	0	0	0	1	84

Fig. 3. Confusion Matrix of the Trained Model

4.3 Model Testing

The model was tested on a set of images that the model has never seen before. A total of 180 images were set aside for this test set. Each class of the classes represented in the training and validation dataset had 10 images per class randomly chosen from the dataset and separated for unbiased results. The prediction has an accuracy of 97.41%. The test images are shown in Fig 4.

Fig 4. Selected Test Results

International Journal of Engineering Applied Sciences and Technology, 2022
Vol. 7, Issue 8, ISSN No. 2455-2143, Pages 166-173
Published Online December 2022 in IJEAST (http://www.ijeast.com)

From the test results shown in the figure above, out of the sampled images, the model failed to classify mango correctly, but it misclassified the image as waterleaf. The test results are acceptable because it classified most of the leaves provided, correctly.

The model was converted to a web useable format by using Tensorflow.js. Figure 5 shows the resulting webpage.

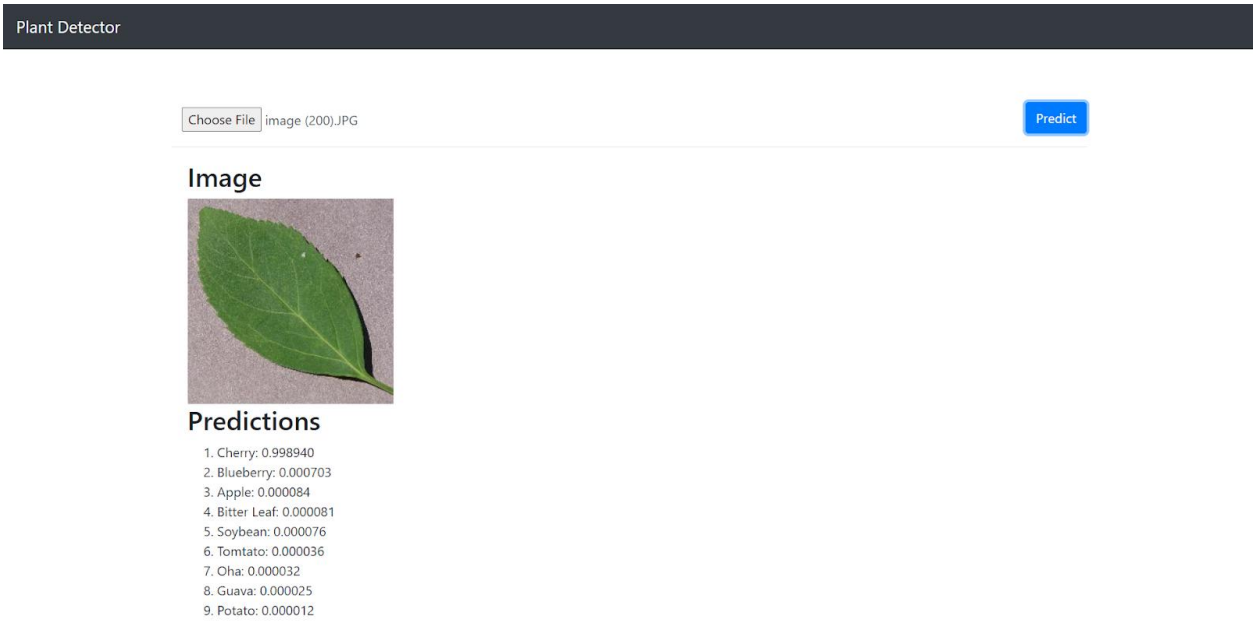

Fig. 5. Prediction Results from an uploaded Image

V. CONCLUSION

This work was able able to successfully train and deploy the MobileNet v1, a lightweight model developed with architectural depth wise separable convolution instead of conventional convolution for plant leaf classification making it easy for smartphones or devices with internet connectivity to use. We achieved this by using the following training parameters, a learning rate of 0.00001, 50 epochs and a batch size of 32 to achieve an accuracy of 97.41%. The use of these parameters trained the MobileNet v1 model to be able to distinguish eighteen different types of plant leaves. The high accuracy of the model could mean that the model is overfitted. Therefore the model might be extracting more parameters from the training dataset than can be justified. Thereby picking noise and irrelevant information that may be present in the data. This could be caused by using a dataset with little variation to train the model. Mode samples may therefore be needed to make this model's prediction more realistic.

VI. REFERENCES

[1] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," SN Comput Sci, vol. 2, no. 3, p. 160, 2021, doi: 10.1007/s42979-021-00592-x.

[2] J. Wäldchen, M. Rzanny, M. Seeland, and P. Mäder, "Automated plant species identification—Trends and future directions," PLoSComput Biol, vol. 14, no. 4, pp. e1005993-, Apr. 2018, [Online]. Available: https://doi.org/10.1371/journal.pcbi.1005993

[3] S. Wu, F. Bao, E. Xu, Y.-X. Wang, Y.-F. Chang, and Q.-L. Xiang, "A Leaf Recognition Algorithm for Plant Classification Using Probabilistic Neural Network," Aug. 2007.

[4] J. Amadi, 2a Adegeye, and V. Alaje, Special Issue: Embracing Science and Technology in Nature Conservation. 2021.

[5] M. U. Ahmad, S. Ashiq, G. Badshah, A. H. Khan, and M. Hussain, "Feature Extraction of Plant Leaf Using Deep Learning," Complexity, vol. 2022, no. 1, p. 6976112, Jan. 2022, doi: https://doi.org/10.1155/2022/6976112.

[6] Kaggle, "Find Open Datasets and Machine Learning Projects | Kaggle." Accessed: May 10, 2025. [Online]. Available: https://www.kaggle.com/datasets

[7] TensorFlow, "TensorFlow Datasets." Accessed: Oct 10, 2022. [Online]. Available: https://www.tensorflow.org/datasets

[8] IPPN, "Plant Phenotyping Datasets." Accessed: Oct 10, 2022. [Online]. Available: https://www.plant-phenotyping.org/datasets-home

[9] Python Software Foundation, "Keras-Preprocessing • PyPI." Accessed: Oct 12, 2022. [Online]. Available: https://pypi.org/project/Keras-Preprocessing/

[10] Python Software Foundation, "Augmentor • PyPI." Accessed: Oct 12, 2022. [Online]. Available: https://pypi.org/project/Augmentor/
[11] O. Russakovskyet al., "ImageNet Large Scale Visual Recognition Challenge," Int J Comput Vis, vol. 115, no. 3, pp. 211–252, Dec. 2015, doi: 10.1007/S11263-015-0816-Y.
[12] Stanford Vision Lab, "ImageNet." Accessed: Oct 11, 2022. [Online]. Available: https://www.image-net.org/challenges/LSVRC/