International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

Adapting Deep-Learning in Early Yam Disease Detection Using Lenet-5 (Adam Optimizer) Convolutional Neural Network Architecture to Improve Productivity and Enhance Farmers Social Habits in the Digital Age

Okimba Peter Etaba[1], Joshua John A. Abubakar[2], Gani Awudu Ishaya[3], Ambituni Enyetere[4]
[1,3,4] Taraba State Polytechnic, Suntai, Nigeria
[2]Ahmadu Bello University, Zaria, Nigeria

ABSTRACT: The agriculture sector faces significant challenges due to diseases affecting crop yields, particularly in yam cultivation. This study explores the adaptation of deep learning techniques for early detection of yam diseases using a LeNet-5 Convolutional Neural Network (CNN) architecture optimized with the Adam optimizer. The fam sides considered are; Ardokola, Zing and Mutum Biu in Taraba State, Nigeria. By leveraging advanced image processing and machine learning methodologies, we aim to develop an effective diagnostic tool that empowers farmers to identify and manage diseases promptly, ultimately improving productivity. This research not only enhances the technological capabilities of farmers in the digital age but also promotes better agricultural practices, fostering social habits that encourage knowledge sharing and community engagement. The proposed system is tested on a comprehensive dataset of yam leaf images, demonstrating its ability to accurately detect various disease conditions at 17.84%. Results indicate a significant improvement in recognition accuracy, suggesting that the integration of AI-driven solutions can transform disease management approaches in yam farming, contributing to sustainable agricultural practices and improved livelihoods for farmers.

KEYWORDS: CNN, Deep Learning, LeNet-5, Yam

1. INTRODUCTION

Yam (Dioscorea spp.) is a crucial staple food crop in many tropical regions, particularly in West Africa, where it plays a significant role in food security and the economy [1]. However, yam production is severely threatened by various diseases, notably yam mosaic virus, anthracnose, and root rot, which can lead to substantial yield losses [2]. The early detection of these diseases is vital for effective management strategies, especially in a landscape where climate change exacerbates agricultural challenges [3].

Over the years, scientific agriculture, also known as conventional or traditional agriculture, applies the use of scientific principles and techniques to improve agricultural productivity and efficiency through scientific research, data analysis, has enhance crop yield, manage pests and diseases, and also optimize resource utilization [4]. This agricultural practice focuses on factors such as soil fertility, irrigation, plant genetics, and pest control, utilizing scientific knowledge to make informed decisions and improve agricultural outcomes [5]. The application of these practices involving use of fertilizers, pesticides, and machinery have to a large extend support and improved crop production in farm lands.

Traditional methods of disease detection often rely on visual inspections, which can be subjective and prone to error. Recent advancements in artificial intelligence, particularly deep learning, have demonstrated great potential in improving the accuracy and efficiency of disease detection [6]. Convolutional Neural Networks (CNNs), specifically architectures like LeNet-5, have been successfully applied in various agricultural contexts, exhibiting expertise in image classification and pattern recognition [7]. The implementation of the Adam optimizer, known for its efficiency in training deep networks, enhances the performance of these models by adapting the learning rate during the training process [8].

Nigeria as the leader in global yam production, reducing losses due to crop diseases will improve her yield and move the world towards food sufficiency in line with the second goal of the UN SDGs. However, traditional crop disease diagnosis is innately influenced by opinions and restricted to countries that can assist and sustain the requisite human framework [9]; [10]. It has been observed that effective disease identification and control leading to maximization of her production capacity ideally puts her on the path to food sufficiency with multiple chain values from its export potential.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

For early disease detection, [11] conceived the machine learning algorithms for on-site classification of both diseased and heathy walnut leaves. The purpose of this study was to build a robust CNN model that is able to classify images of leaves, depending on whether or not these are infected by anthracnose, and therefore determine whether a tree is infected.
However, focus is on disease identification for yam through image detection and processing of disease sites such as leaves, tuber and root. Deep learning applications in agriculture promises early and accurate identification of crop diseases (up to 98%) [12]; [13] than any other known conventional methods. With the aid of high mobile phone penetration in rural areas in Nigeria [14], deploying DRL application using mobile phone platforms would significantly make yam disease identification easier and faster.
In this digital age, adopting AI-driven solutions not only fosters agricultural productivity but also enhances farmers' social habits by encouraging the integration of technology into farming practices. This research aims to adapt the LeNet-5 architecture for the early detection of yam diseases, thereby providing farmers with a reliable tool that improves their decision-making processes and promotes proactive disease management. By enabling farmers to promptly identify and address disease outbreaks, we aim to contribute to greater yield stability, sustainable farming practices, and enhanced community engagement within the agricultural sector.

2. METHODOLOGY

The methodology for adapting deep learning in early yam disease detection using the LeNet-5 architecture optimized with the Adam optimizer consists of the following phases:

1. Data Collection

A comprehensive dataset of yam leaf images was collected from various farm sides in Ardokola, Zing and Mutum Biu, Taraba State, encompassing white yam at various stages of healthy and diseased of the plant. This dataset includes images affected by common diseases such as yam virus disease, anthracnose, mechanical damage, and tuber disease, ensuring a diverse representation. Images were sourced from local farms, agricultural research centers, and publicly available datasets to build a robust training and testing dataset.

2. Data Preprocessing

The collected images were subjected to several preprocessing steps to enhance the model's performance:

i. Image Augmentation: To increase the diversity of the dataset and improve the model's generalization abilities, techniques such as rotation, flipping, scaling, zooming, and colour adjustment were applied.
ii. Resizing: All images were resized to a uniform dimension (e.g., 32 x 32 pixels) to match the input requirements of the LeNet-5 architecture.
iii. Normalization: Pixel values were normalized to a range between 0 and 1 to facilitate faster convergence during model training.

3. Model Architecture

The LeNet-5 architecture was implemented as follows:

i. Input Layer: Accepts images of size 224 x 224 pixels with three color channels (RGB).
ii. Convolutional Layers: The model consist of two convolutional layers (Conv1 and Conv2) with corresponding activation functions (such as ReLU).
iii. Pooling Layers: Max-pooling layers was employed after each convolutional layer to reduce dimensionality and extract significant features.
iv. Fully Connected Layers: The feature maps were flattened and fed into two fully connected layers for classification.
v. Output Layer: The output layer used a SoftMax activation function to classify the images into predefined categories: healthy and various diseased classes.

4. Model Training

The training process involve the following steps:

i. Splitting the Dataset: The dataset was divided into training (70%), validation (15%), and testing (15%) sets to evaluate the model's performance adequately.
ii. Loss Function: Categorical cross-entropy was used as the loss function for multi-class classification.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

iii. Optimizer: The Adam optimizer was utilized to adjust the learning rate dynamically, allowing for faster convergence.
iv. Batch Size and Epochs: The model was trained using a batch size of 32 and for a predetermined number of epochs (such as., 6, 8, 10, 12) while monitoring the validation loss for early stopping.

5. Model Evaluation

The trained model was evaluated based on performance metrics, including:

i. Accuracy: The percentage of correctly classified images in the test set.
ii. Precision, Recall, and F1_Score: To assess the model's performance specifically on diseased categories.
iii. Confusion Matrix: To visualize the model's classification capabilities and identify areas for improvement.

4. INPUT/OUTPUT FORMAT

A. Input format

The inputs to the CNN are images collected from farm sites where yams are cultivated. The images of diseases affecting the popular types of yams in these areas were collected using normal cameras as well as phone cameras. Note that after preprocessing, an image size of 224 x 224pixels were fed batch per iteration to the CNN architecture (LeNet-5). Figure 1 depicts a diseased leaf of yam while Figure 2 shows a collection of several leaves. Figure 3 shows a collection of tuber disease captured from storage barns. Figure 4 demonstrate how the input process occurs from dataset to the finished input shape called the pixel.

Figure 1: A Yam disease infected leaf

Figure 2: A colony of Yam diseased leaves

Figure 3: Collection of diseased tubers

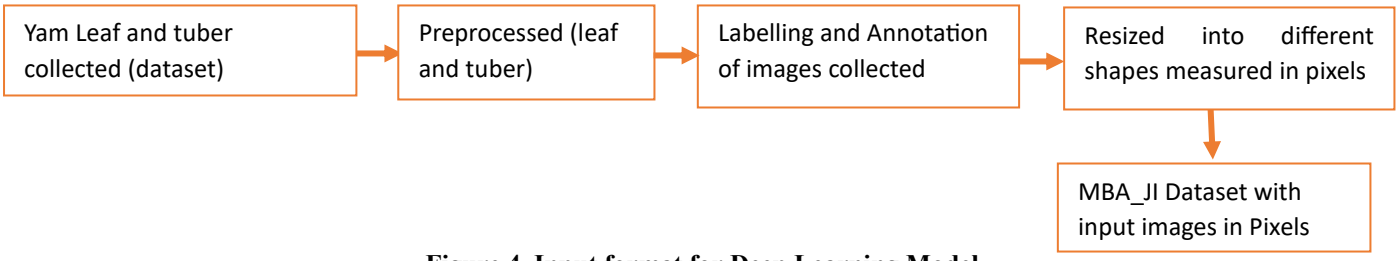


Figure 4. Input format for Deep Learning Model

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

B. Output format

The output of the system were the accuracies generated by testing the performance of CNN architecture and several properties of the confusion matrix, which include False Negative Rate (FNR), False Positive Rate (FPR), True Positive Rate (TPR) and True Negative Rate (TNR). Output format is as presented in Receiver Operating Characteristics (ROC) curves as well as Confusion matric.

5. ALGORITHM

Here, the pseudocode for the simplest deep learning neural network was presented so as to show what was basically expanded in the program codes. Summarily, the Google Colab notebook allows the import of several libraries, the definition and compiling of the CNN, the provision of data, the training of the network and the prediction/classification of the diseases. On CNN architecture, code of its implantation on TensorFlow were included.

1. Import libraries
   import tensorflow as tf
   import numpy as np
   from matplotlib import pyplot as plt
   from tensorflow import keras
   from tensorflow.keras.applications.lenet-5 import LeNet-5
   from tensorflow.keras.applications.lenet-5 import preprocess_input
   from tensorflow.keras.applications.lenet-5 import LeNet-5
2. Mount the drive paths for training, testing and validation
3. Declare paths for training, testing and validation.
4. Data preparation
   Read images into declared paths and returns a numpy array for images
   Resize the image array to acceptable formats for a specific CNN
5. Define and Compile the Convolutional Neural Network
   model = tf.keras.Sequential([keras.layers.Dense(units=1, input_shape=[1])])
   model.compile(optimizer='sgd', loss='mean_squared_error')
6. Providing the Data
   xs = np.array([ ], dtype=float)
   ys = np.array([ ], dtype=float)
7. Applying CNN Architectures
   LeNet-5
   lenet-5 = LeNet(input_shape=(224, 224, 3), weights='imagenet', include_top=False)
   Do not train the pre-trained layers of LeNet-5
   x = Flatten()(vgg.output)
   Add output layer.Softmax classifier is used as it is multi-class classification
   Prediction = Dense(3, activation='sigmoid')(x)
   model = Model(inputs=vgg.input, outputs=prediction)
8. Compile the model
   Add the Adam optimizer
   Add the sparse_categorical_crossentropy loss function
   Add the learning rate
   Add the metrics=['accuracy'])
9. Training the Neural Network
   Fit the model and specifying the training epochs
   Print the model predictions
10. Plot Training and Validation Accuracy

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

11. Plot Training and Validation Loss for the CNN Architectures
12. Generate the classification report
13. Generate the classification metrics

6. LeNet-5 CNN Architecture

The LeNet-5 CNN architecture [7] comprises a total of 7 layers (Figure). These layers consist of 3 convolutional layers, 2 subsampling (or pooling) layers, and 2 fully connected layers. The first layer, known as the input layer, does not contribute to learning as it simply takes in the input data. Specifically, the LeNet-5 model is designed to handle images with dimensions of 32 x 32, which serve as the input dimensions for the subsequent layer in the network. Also, LeNet gave an accuracy of 98% here (Alake, 2020). While applying the LeNet-5 CNN Architecture to the yam dataset, Figure 5 showed the plot of Training Accuracy and Validation Accuracy, Figure 6 showed the plot of training loss and validation loss, while Figure 7 depicts the confusion matrix.

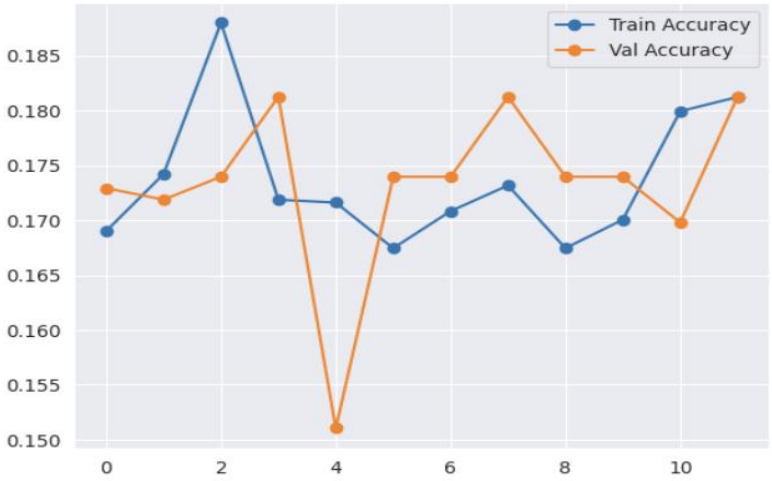


Figure 5: LeNet-5 CNN Architecture (Adam Optimizer) plot of Training Accuracy and Validation Accuracy

When the validation accuracy and training accuracy converge at some points, it indicates that the model is performing consistently well on both the training and validation datasets as it is seen in the ROC curve above. This convergence suggests that the model is not overfitting the training data and is able to generalize effectively to unseen data. It shows that the model is learning the underlying patterns in the data and is making accurate predictions on both the training and validation sets. This convergence is a positive sign that the model is performing well and has learned to generalize effectively with accuracy of 17.84%. The trainable parameter used is 46,661,506 with an input shape of 224 x 224 pixels at 12 epochs applying early stopping.

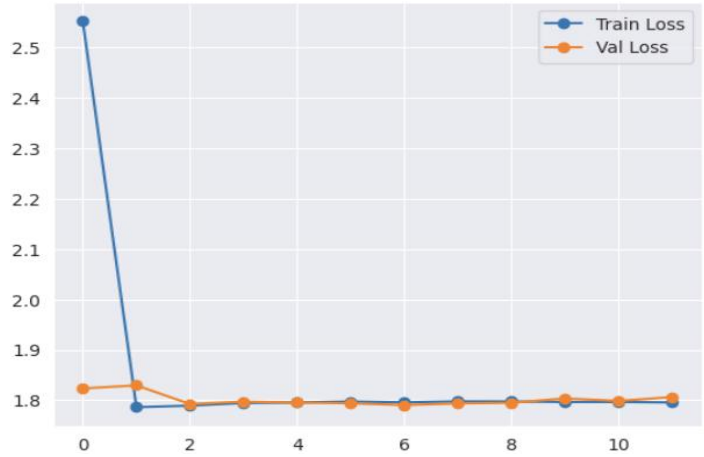


Figure 6: LeNet-5 CNN Architecture (Adam Optimizer) plot of Training Loss and Validation Loss

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

In the context of the Receiver Operating Characteristic (ROC) curve for LeNet-5 with Adam Optimizer as illustrated above, the validation loss and training loss converges at some points, meaning that the model has learned to generalize well to unseen data. This convergence indicates that the model is not overfitting the training data and is able to make accurate predictions on new, unseen data. It suggests that the model has reached a good balance between fitting the training data well and being able to generalize to new data points.

Figure 7: Confusion Matrix of LeNet-5 CNN Architecture (Adam Optimizer)

The classification report provided presents the performance metrics of a classification model. Let's interpret the report class by class. For Class 0, there was no samples, so precision, recall, and F1-score are all reported as 0. Since there are no samples, there is no support value.

Class 1: Similar to Class 0, this class has no samples, resulting in precision, recall, and F1-score all being reported as 0. The support value is also 0.

Class 2: Similar to Classes 0 and 1, this class has no samples, resulting in precision, recall, and F1-score all being reported as 0. The support value is 0.

Class 3: This class has 1191 samples. The precision for this class is reported as 0.99, indicating that when the model predicts samples belonging to this class, it is correct 99% of the time. The recall is reported as 0.18, which means that only 18% of the true positive samples for this class are correctly identified by the model. The F1-score, which combines precision and recall, is 0.30. The support value represents the number of samples in this class.

Class 4: This class has only 3 samples. The precision, recall, and F1-score for this class are all reported as 0, indicating that the model did not correctly identify any samples belonging to this class. The support value is 3.

Class 5: Similar to Classes 0, 1, and 2, this class has no samples, resulting in precision, recall, and F1-score all being reported as 0. The support value is 0.

Overall, the accuracy of the model is reported as 0.18, meaning that it correctly predicted the class labels for approximately 18% of the samples. The macro average F1-score is 0.05, indicating a low overall performance. The weighted average F1-score is 0.30, which takes into account the class imbalance. However, given the low performance on most classes and the absence of samples in some classes, it is important to assess the model's performance critically and consider further improvements.

7. RESEARCH FINDINGS AND IMPLICATIONS

Pre-trained CNN Architecture for Training Accuracy and Validation Accuracy

This section presents and discusses the findings and implication of the study in the pre-trained CNN architecture. The performance of the CNN architecture used is as discussed;

The CNN used was LeNet-5 with Adam Optimizer acting on a total of 46,661,506 parameters It representing the overall complexity and size of the model was distributed as;

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

Trainable parameters - 46,661,506
Non-trainable parameters - 0

Other parameters used include; Training Time, Input Shape, Early stopping and Accuracy text. The "Trainable Parameters" shows the number of parameters that are updated during the training process. These parameters are learned and optimized based on the given dataset. The "Non-trainable Parameters" represents the number of parameters that are not updated during the training process. These parameters are typically pre-trained or fixed and contain valuable knowledge transferred from previous training. Training Time (Mins) provides the training time in minutes for the CNN architecture. The training time indicates the duration required to train the model on the given dataset. The "Input Shape" describes the dimensions of the input images expected by the CNN architecture. This was; 224 by 224 applying LeNet-5. The "Early Stopping" indicates whether early stopping was applied during the training process. Early stopping is a technique to prevent overfitting by stopping the training if the performance on a validation set starts to degrade. The values in this column indicate "Yes (Epochs X)," where X represents the number of epochs at which early stopping was triggered. The last "Accuracy (Test set)," shows the accuracy achieved by the CNN architecture on the test set. The value was 17.84% in using LeNet-5 with Adam Optimizer at a training time 22mins 49sec applying early stopping at 12 epochs. This value demonstrated that yam plant disease can be moderately predicted at early symptoms to mitigate against possible spread to other plant (Yam), thus architectural design decision of 224 by 224 pixel greatly affected the complexity and performance of the model.

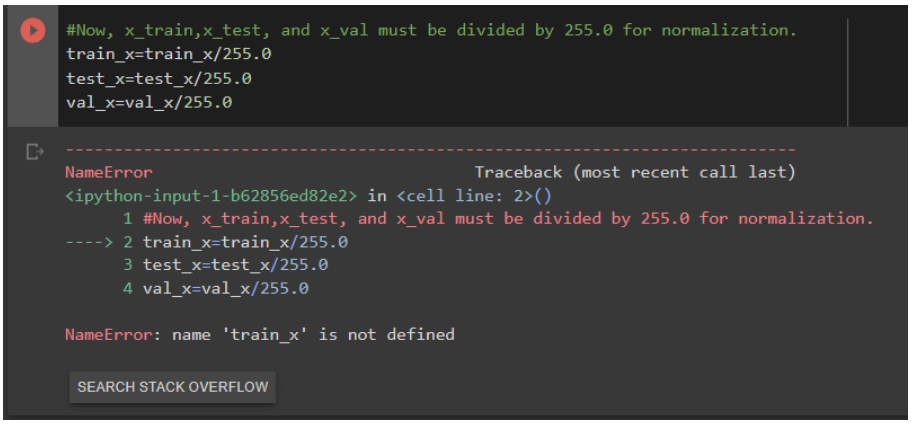
Pre-trained CNN Architecture for Training Loss and Validation Loss

For LeNet-5 with the Adam optimizer as shown in figure 6, the training loss was relatively high (1.7956), and the training accuracy was low (0.1812). The validation loss and accuracy were similar to the training metrics, with a validation loss of 1.8064 and a validation accuracy of 0.1813. On the test set, the model achieved a test loss of 1.8077 and a test accuracy of 0.1784.

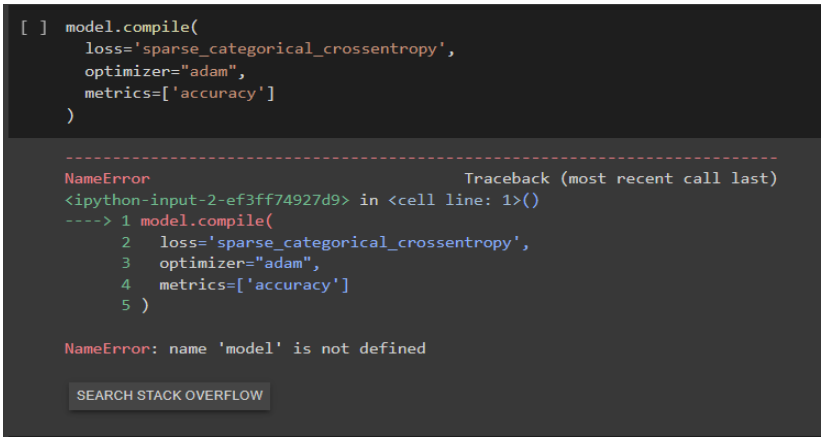
8. LIMITATIONS OF MODEL IMPLEMENTATION

Since this is an image classification project, it is expected that there would be need for high use of random-access memory (RAM), which is another beautiful reason for the use of Graphical Processing Unit (GPU) functionality in Google Colab.

Runtime Disconnection – This error kept occurring at the data preparation stage, and the error description is as follows: "You are connected to a GPU runtime, but not utilizing the GPU". Note that at this stage, the cv2.imread() function from OpenCV library was used to read images from a file. Additionally, it takes the file path as input and returns a NumPy array representing the image. Then, the img_arr variable in your code is assigned the NumPy array representing the image read from the image_path file. Whenever, the runtime is disconnected it presents the errors, for instance the below codes for model not defined for train_x and model are depicted as Figure and Figure. Essentially, subscribing to Colab Pro user and enabling GPU through the following actions: go to upper toolbar > select 'Runtime' > 'Change Runtime Type' > hardware accelerator: select 'GPU'. Also, the python module called Pickle was also of great help at this point. Example of the messages are as shown;



International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org



Herdsmen Attack on Farms and Farmlands – Herdsmen attacks on farms and farmers in Nigeria have significant consequences, impacting various aspects of society, economy, and security. Some of the consequences include: Loss of Lives and Injuries and Destruction of Crops and Farmlands. For the former, attacks often result in the loss of lives and injuries to farmers and local communities. These attacks can lead to violent clashes, resulting in casualties and the displacement of people from their homes. On the latter, attacks involve the intrusion of cattle into farmlands, leading to the destruction of crops, such as maize, rice, yam, and other agricultural produce. This not only causes immediate economic losses but also disrupts food production and security in affected areas. These has made it grossly difficult to go these farmlands for collection of more images.

9. DEPLOYMENT AND COMMUNITY ENGAGEMENT AND TRAINING

Upon successful evaluation, the trained model is to be deployed as a user-friendly application or web interface that allows farmers to upload images of yam leaves for disease diagnosis. The application will provide immediate feedback along with recommendations for managing detected diseases.

To enhance farmers' social habits and ensure effective utilization of the technology, workshops will be organized to train farmers on the use of the diagnostic tool. This phase will include demonstrations, hands-on practice, and discussing integrated disease management practices.

10. FEEDBACK AND ITERATION

Regular feedback will be collected from the farmers using the application to identify challenges and areas for improvement. The model will be iteratively refined based on this feedback to enhance user experience and diagnostic accuracy.

11. SYSTEM SECURITY

When it comes to system security in a deep learning project for yam disease classification, there are several considerations to keep in mind. By the consideration of these aspects of system security, the integrity, confidentiality, and availability of the deep learning system for crop disease classification are protected, thus ensuring the trust and safety of the users and their data. Here are some key points:

Data Security: Protecting the data used for training the model is crucial. Ensure that sensitive data, such as farmer or user information, is properly stored securely. Implement access controls to limit data access to authorized individuals only.

Model Security: Safeguard the trained model to prevent unauthorized access or tampering. Consider using encryption techniques to protect the model during storage and transmission. Implement secure protocols for model deployment and ensure that only authorized parties can access and utilize the model.

Secure Communication: If the deep learning system communicates with external entities or services, such as APIs or databases, ensure that secure communication protocols (e.g., HTTPS) are implemented to protect the confidentiality and integrity of the data exchanged.

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

Authentication and Authorization: Implement strong authentication mechanisms to verify the identity of users or systems accessing the deep learning application. Use role-based access controls to grant appropriate privileges to different user roles, ensuring that only authorized individuals can interact with the system.
Secure Infrastructure: Pay attention to the security of the underlying infrastructure where the deep learning system is hosted. Keep software frameworks, libraries, and operating systems up to date with the latest security patches. Employ firewalls, intrusion detection systems, and other security measures to protect against potential threats.
Monitoring and Logging: Set up monitoring and logging mechanisms to track system activities, detect anomalies, and identify potential security breaches. Regularly review logs and investigate any suspicious activities or access attempts.
Privacy and Compliance: Ensure compliance with applicable privacy regulations and standards, such as GDPR or HIPAA, depending on the nature of the data being processed. Implement privacy protection measures, such as data anonymization or aggregation, where necessary.
Regular Audits and Assessments: Conduct regular security audits and assessments of the deep learning system to identify vulnerabilities, evaluate risks, and implement necessary security enhancements. This includes penetration testing, code reviews, and vulnerability scanning.
User Awareness and Training: Educate users, administrators, and developers involved in the deep learning project about security best practices, including password hygiene, social engineering awareness, and safe coding practices.

CONCLUSION

First of all, the study involved the collection of images of diseases (virus) and grasshopper attack, and mechanical damages on yam leaves and tubers from selected farm sites in Taraba State, Nigeria. Besides image capturing or collection, specific objectives included performing model or CNN architecture modifications, hyperparameter tuning, and implementing simulation experiments using this architecture. Second, the work examined an existing research effort in this subject to see where and how new knowledge may be contributed. Investigating the extensive literature on disease classification, information (knowledge) gaps were identified as a result of the review, which gave the justification for conducting the research. Thereafter, methodological considerations and attendant steps were discussed, the actual implementation of the CNN model and architectures for yam diseases classification was done. Specifically, Pre-trained CNN Architecture such as LeNet-5 (Adam Optimizer), was employed for classification. By following this comprehensive methodology, an effective deep learning-based solution was developed for early yam disease detection, improving agricultural productivity and fostering a more connected farming community in the digital age.

REFERENCES

1. FAO. (2022). The State of Food Security and Nutrition in the World. Food and Agriculture Organization of the United Nations. Retrieved from [FAO Website] (http://www.fao.org/publications/sofi/en/)
2. Odebode, A. C., Fagbohun, E. D., & Awe, K. K. (2020). Integrated Management of Yam Mosaic Virus Disease. Journal of Agricultural Research, 15, 18-25.
3. Khan, A., Adnan, M., & Bhatti, M. (2021). Climate Change and Crop Diseases: A Review. Plant Disease, 105(6), 1343-1357. doi:10.1094/PDIS-10-20-2242-FE
4. Pivoto D, Waquil PD, Talamini E, Finocchio CPS, Dalla Corte VF, de Vargas Mores G (2018) Scientific development of smart farming technologies and their application in Brazil.Information Process Agriculture (5) pp21–32.
5. Hemming J, Ruizendaal J, Hofstee J, Van Henten E, Hemming J, Hofstee J. W. (2014). Fruit detectability analysis for different camera positions in sweet- pepper. Sensors (14) pp 6032–6044. https://doi.org/10.3390/s140406032
6. Mohammed, F., Kandeh, S., & Abubakar, S. (2022). Leveraging Deep Learning for Crop Disease Detection: A Review. Agricultural Systems, 195, 103313. doi:10.1016/j.agsy.2021.103313
7. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11), 2278-2324. doi:10.1109/5.726791
8. Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980.
9. Bock, C., Poole, G., Parker, P., and Gottwald, T. (2010). Plant disease severity estimated visually, by digital photography and image analysis, and by hyperspectral imaging. *Crit. Rev. Plant Sci.* 29, 59–107. doi: 10.1080/07352681003617285

International Journal of Current Science Research and Review
ISSN: 2581-8341
Volume 08 Issue 01 January 2025
DOI: 10.47191/ijcsrr/V8-i1-24, Impact Factor: 7.943
IJCSRR @ 2025
www.ijcsrr.org

10. Ramcharan, A., McCloskey, P., Baranowski, K., Mbilinyi, N., Mrisho, L., Ndalahwa, M., Legg, J. and Hughes, D. P. (2019). A Mobile-Based Deep Learning Model for Cassava Disease Diagnosis. Frontiers in Plant Science, Volume 10 (272), pp 1- 8. DOI: 10.3389/fpls.2019.00272
11. Anagnostis, A., Asiminari, G., Papageorgiou, E. and Bochtis, D. (2020). A Convolutional Neural Networks Based Method for Anthracnose Infected Walnut Tree Leaves Identification. *Applied Sciences*, 10(469). DOI: 10.3390/app10020469.
12. Ferentinos, K. P. (2018). Deep learning models for plant disease detection and diagnosis *Computers and Electronics in Agriculture*, 145, 311 – 318. DOI: 10.1016/j.compag.2018.01.009
13. Singh, A. K., Ganapathysubramanian, B., Sarkar, S. and Singh, A. (2018). Deep Learning for Plant Stress Phenotyping: Trends and Future Perspectives. *Trends in Plant Science*. October 2018, Vol. 23, No. 10, pp. 883 - 898. https://doi.org/10.1016/j.tplants.2018.07.004
14. The Guardian Newspaper. 16th March, 2018. https://guardian.ng/business-services/nigerias-mobile-phone-penetration-hits-84-per-cent/

Cite this Article: Etaba O.P., Abubakar J.J.A., Ishaya G.A., Enyetere A. (2025). Adapting Deep-Learning in Early Yam Disease Detection Using Lenet-5 (Adam Optimizer) Convolutional Neural Network Architecture to Improve Productivity and Enhance Farmers Social Habits in the Digital Age. International Journal of Current Science Research and Review, 8(1), 235-244, DOI: https://doi.org/10.47191/ijcsrr/V8-i1-24