Distributed Deep Learning for Maize Disease Detection

Basit Hussain
Brac University
Dhaka, Bangladesh

Christian Boateng
Brac University
Dhaka, Bangladesh

Annajiat Alim Rasel
Brac University
Dhaka, Bangladesh

Abstract

Maize, a critical staple crop, faces significant challenges from diseases like Maize Lethal Necrosis (MLN) and Maize Streak Virus (MSV), which threaten food security and economic stability in Sub-Saharan Africa. Many diagnostic techniques are time-consuming, expensive, or unavailable to smallholder farmers, making efficient, scalable and accurate diagnostic techniques important. This study introduces a distributed deep learning framework leveraging the EfficientNetV2B2 architecture and TensorFlow's MultiWorkerMirroredStrategy to enhance maize disease detection. Utilizing a dataset of over 17,000 labeled maize leaf images. Experiments were conducted on single and distributed setups using Intel Core i7 systems, showing that the framework is able to operate in resource limited environments. The distributed model was able to detect with a 92% accuracy, and required 30% less training than the centralized model. This research demonstrates how distributed deep learning is feasible as methods that can transform agricultural diagnostics and disease prevention, and contribute to food security.

CCS Concepts

• **Do Not Use This Code → Generate the Correct Terms for Your Paper**; *Generate the Correct Terms for Your Paper*; Generate the Correct Terms for Your Paper; Generate the Correct Terms for Your Paper.

Keywords

Distributed Deep Learning, Maize Disease Detection, Food Security, EfficientNetV2B2

ACM Reference Format:

Basit Hussain, Christian Boateng, and Annajiat Alim Rasel. 2025. Distributed Deep Learning for Maize Disease Detection . In . ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Cultivation is a central component of food security and economic stability on the global level [6]. Among the staple crops, Maize (Zea mays) locally called corn has been a source of staple foods in diets. It is widely grown throughout Sub-Saharan Africa, but especially in Tanzania: there are more than 5 million hectares of maize cultivated, and the population consumes 128 kg of it per year on average. Apart from food, it is used for feeding animals, producing fuel, and making biodegradable plastics [2].

Nonetheless, maize production comes with several challenges, in particular several diseases including Northern Leaf Blight, Common Rust, Gray Leaf Spot, Maize Lethal Necrosis (MLN), and Maize Streak Virus (MSV). Among these, MLN and MSV are two devastating diseases that pose a significant threat to maize crops, particularly in Sub-Saharan Africa.

MLN, due to maize chlorotic mottle virus (MCMV) and sugarcane mosaic virus (SCMV) leads to chlorosis, necrosis, stunting and plant death, with subsequent yield reductions. MSV that is transmitted by the leafhoppers leads to leaf shedding and poor quality grains. Although, the outbreaks of both diseases have been noted in different countries in Africa such as Kenya, Ethiopia, Nigeria, Tanzania, Uganda, Rwanda among others. These diseases have serious socio-economic effects, they lower yields, increase prices and pose a threat to food security. The current methods of assessment are either slow or are very costly to the small holder farmer, hence the need for more accurate and efficient methods of assessment.

Previous studies have utilized modern approaches in AI, more specifically deep learning with CNNs [1], as well as machine learning models, to identify diseases affecting maize crops. Where CNN models have exhibited higher accuracy in disease diagnosis than the conventional machine learning models. Recent developments in deep learning have given hope in the accurate diagnosis of plant diseases. However, large scale training of deep neural networks is computationally expensive, and centralized approaches are not feasible in scenarios with low bandwidth. In addition, privacy becomes an issue when sharing agricultural data between different regions.

These challenges are addressed in this research by proposing a distributed deep learning framework for maize disease detection with explainable AI to improve model interpretability.

2 Related Work

A lot of research has been done on using deep learning methods for agriculture diagnosis, where architectures like CNNs and transfer learning architectures to plant diseases detection have been presented. Some recent works have looked at the issues of scaling the training process and the privacy concerns that are associated with such processes as distributed training and federated learning. Nevertheless, there is a lack of research on the integration of distributed systems with Explainable AI. This work further extends these developments by proposing a solution that incorporates both distributed training and XAI.

This study [7] utilized Federated Learning with EfficientNet-B0 to bridges privacy preservation and medical imaging. It has high diagnostic accuracy and privacy and does not allow data heterogeneity like ResNet. Despite the limitations such as the weak interpretability and generalization, the research sets a solid groundwork

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

for incorporating privacy preserving AI into further healthcare diagnostics, that will bring revolutionary changes to medical imaging and privacy compliance.

In the present study [3], the authors introduce an enhanced Federated Deep Learning (FDL) model for plant leaf diseases classification that solves the issues associated with centralized learning, including; data imbalance, expensive computation, and the inconvenience of data collection. The proposed FDL allows local models to train datasets of a particular region and share only the weights of the model with the parent model, thus having minimal computational needs. The authors propose a light and efficient Hierarchical Convolutional Neural Network (H-CNN) which has 0.09 million parameters and 0.35 MB model size for both parent and child model. On the Cherry, Maize, and Tomato dataset as well as the Apple, Maize, and Tomato dataset, the FDL was proven superior to centralized learning as well as advanced federated techniques such as FedAdam and FedAvg with testing accuracies of 98% and 93% respectively. Thus, this work shows that the proposed framework can be used for large-scale, resource-constrained, and accurate identification of plant diseases in real-world settings despite such limitations as working with asynchronous strategies and noisy data.

The study [5], proposes a Federated Learning CNN model for the identification of maize diseases, based on decentralized and sensitive data. The model had the overall accuracy of 89.4% better than traditional CNNs and other ML algorithms metrics in terms of precision, recall, and F1-score. It showed robustness across different disease classes and regions, which underlined the applicability of Federated Learning for efficient crop disease control.

In this study [4], PyTorch Distributed Data Parallel (DDP) module is designed to solve the problem of scalability and equivalence. It maintains mathematical consistency with local training while achieving near-linear scalability across large-scale systems. For performance optimization, DDP includes a variety of techniques, such as gradient bucketing, which increases communication efficiency by grouping gradients into larger units; and by overlapping communication and computation, reducing latency by synchronizing gradients during the backward pass. In addition, skipping gradient synchronization minimizes unnecessary communication overhead. Real-world impact: DDP has broad internal adoption at Facebook and more generally across the open-source community, where it has proven to be extremely effective and versatile for a wide range of deep learning applications.

3 Methodolgy

3.1 Dataset Preparation

The data collection process involved acquiring maize leaf images from farmers' gardens in Tanzania using the AdSurv mobile application installed on Samsung phones.

A group of researchers and students from Tanzania Agricultural Research Institute and The Nelson Mandela African Institution of Science and Technology collected the dataset for a period of six months, from February 2021 to July 2021. The images were collected to diagnose MLN and MSV diseases as shown in figure 1, aiming to assist farmers in disease diagnosis and improve maize production.

The dataset consists of 17,277 labeled images categorized into Healthy (5,542), Maize Lethal Necrosis (5,068), and Maize Streak Virus (6,667) as depicted by figure 2. Each image instance includes the crop status, variety, age, and location (district, sub-county). The data collected is well-labeled and curated, providing an open and accessible maize image dataset for machine learning experiments.

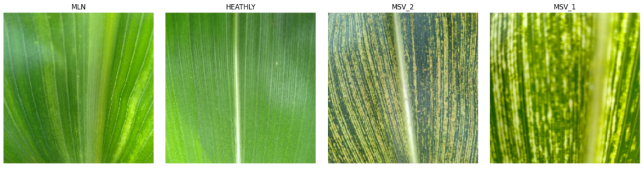


Figure 1: Sample images from the dataset categorized as Healthy, MLN, and MSV.

Figure 2: Distribution of dataset categories. (A bar chart showing the count of Healthy, MLN, and MSV images to be included here)

3.2 Model Architecture

The proposed framework leverages the EfficientNetV2B2 architecture for its exceptional efficiency and performance in image classification tasks. Developed by Mingxing Tan and Quoc V. Le, EfficientNetV2B2 is a convolutional neural network (CNN) that excels in faster training speeds and greater parameter efficiency. These attributes make it particularly well-suited for distributed deep learning systems operating in resource-constrained environments.

EfficientNetV2B2 was pre-trained on ImageNet and used as the backbone for feature extraction. The top classification layers were removed to allow for customization, enabling the model to specialize in maize disease classification. Custom layers were added to adapt the model, including a Global Average Pooling Layer, a Fully Connected Dense Layer with 256 neurons and ReLU activation, a Dropout Layer for regularization (rate = 0.5), and an Output Layer with three neurons (softmax activation) to classify Healthy, MLN, and MSV classes.

To further enhance efficiency, the pre-trained weights of EfficientNetV2B2 were frozen during the initial training phase. This

approach allowed the model to leverage previously learned features, reducing computational overhead and improving convergence. These features collectively make the framework robust and efficient for agricultural diagnostics.

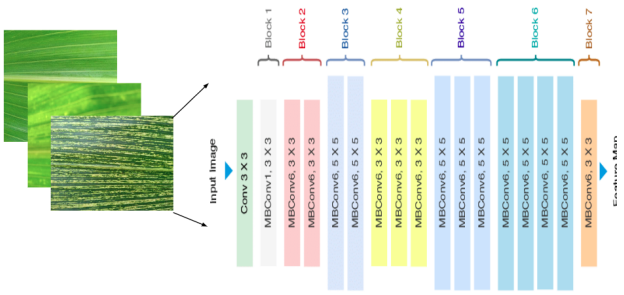


Figure 3: EfficientNetV2 Structures.

3.3 Distributed Training Framework

Distributed training using two Core i7 laptops was set up for faster training and better computational efficiency. The laptops were configured in a master-worker architecture leveraging TensorFlow's MultiWorkerMirroredStrategy

Table 1: Configuration Details of the Distributed Training Framework

Component	Details
Master Laptop	
Processor	Core i7
RAM	16GB
OS	Windows 10
Libraries	TensorFlow, NumPy, Pandas, Matplotlib, Scikit-learn
Worker Laptop	
Processor	Core i7
RAM	16GB
OS	Windows 10
Libraries	TensorFlow, NumPy, Pandas, Matplotlib, Scikit-learn

The environment setup process started with making both the Laptops equally configured and the deployment of basic libraries like TensorFlow, NumPy, Pandas, Matplotlib and Scikit-learn. To get the synchronous environment for the interaction of the two devices, all the data was shared between the two laptops through file sharing.

With the cluster setting, the master laptop was expected to coordinate the training process while the worker was to perform computation. For insistence, when the application required that information be transferred from one laptop to the other, their IP addresses were established and a channel for data transfer was created.

Since data augmentation was used, ImageDataGenerator class in TensorFlow was used to configure the data pipeline. This included real time transformations like rotation, scaling and flipping so that the model would locate the inputs in more ways. The data directory for both laptops was kept the same at all the stages of the process to reduce variability of the results.

During the model training, the model was wrapped inside the scope of the strategy so that it will synchronize as needed. The use of fit method in training was implemented such that TensorFlow could distribute the data between the two laptops. TensorFlow synchronized the weights and gradients across two nodes of computation, thereby keeping the distributed model learning efficient.

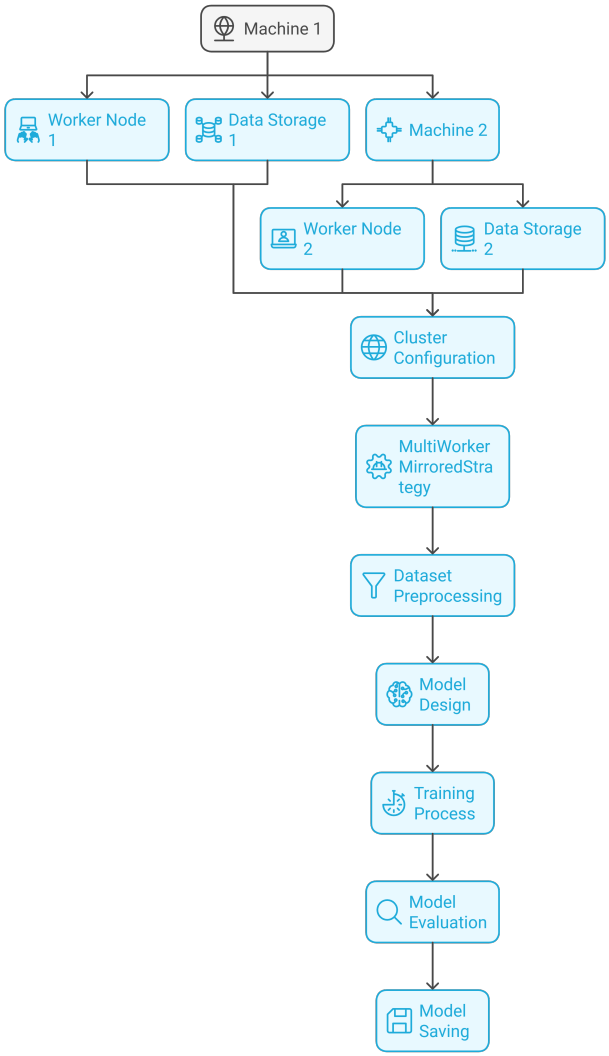


Figure 4: Distributed System Architecture

This flowchart, shown in fig. 4, gives a detailed description of the distributed training process, clarification of different system set

up, data sharing, and the process of model optimization from the master to worker.

3.4 Training on a Single Core i7 System

The model training was done on a single Intel Core i7 laptop. The computational environment was set up with TensorFlow and primary libraries including NumPy, Pandas, and OpenCV to facilitate the run of the training process. In the setup, the elements targeted for computational enhancement were memory owing to the constrained hardware resources available.

The maize image dataset was collected locally, and images in it were further divided into training sets, validation sets, and test sets. Real-time preprocessing and augmentation were performed using TensorFlow utilities. To prepare the images, they were resized to match the EfficientNetV2B2 image size of 224×224, and the pixel intensity values were normalized to 0-1 range; other preprocessing steps used included; rotation, scaling, and horizontal flip. All these transformations were limited to the training set in order to generalize the results, since the validation and test set used remained constant. For the purpose of reproducing the data directory structure remained consistent throughout the whole project.

The EfficientNetV2B2 model was fine-tuned from a model initialized with weights trained on the ImageNet dataset. The layers of the base model were modified with new top layers that were particular to maize image classification. Others were a global average pooling layer, a dense layer comprising 256 neurons with ReLU activation function, a dropout layer with regularization, and a final softmax layer for multi-class classification.

Training was implemented with a small batch size in order to minimize the memory. The fit method in TensorFlow was used for training with the training dataset so that iterative updates of model weights could be given. To evaluate the converging speed, validation was carried out at each epoch to check the improvement done. In this context, the early stopping criterion was used to stop the training process after the maximum of cross-entropy on the validation data set reached its optimum point. The whole pipeline was medium-large in terms of computational complexity but optimized to be end-to-end for model-making on a single Core i7 system thus providing a rapid solution to the problem.

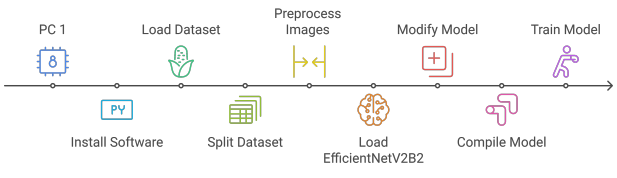


Figure 5: Single PC Training Architecture

The flowchart in fig. 5, provides a detailed overview of the training pipeline, illustrating each step from data preprocessing to model optimization.

4 Experimental Setup

The experiment was designed to evaluate the performance of the EfficientNetV2B2 model on a maize image dataset, leveraging two different computational setups: Two Intel Core i7 laptops, one Core i7 laptop, and a distributed system. All of the configurations had 16GB of RAM and Windows 10 as the operating System. The big difference between the two setups is how training is executed and how data is distributed.

4.1 Single Core i7 System Setup

The training in the initial phase of the experiment was done on a single laptop with intel Core i7 processor. For model retraining, we had set up TensorFlow and Keras in the software environment running on Python 3.x. Furthermore, the libraries relevant for said types of tasks such as handling data preprocessing (NumPy, Pandas, and OpenCV) were included. The maize image dataset was stored locally on the laptop and was split into train, validation and test subsets. Image resizing was used as data prep steps and resized the images to 224 × 224 pixels in order to match the EfficientNetV2B2 architecture, normalizing pixel values to the [0, 1] range, apply augmentation techniques like rotation, scaling and flipping on the training data. In order to compare to other DBNs, performance evaluation was performed using unchanged validation and test data.

However, training on the dataset was feasible as the Intel Core i7 processor had no dedicated GPU, we had to rely on the CPU processing power. The same model was trained with TensorFlow fit method by reducing the batch size so that memory usage would be minimized. Early stopping mechanisms and a learning rate scheduler were implemented to reduce overfitting and the possible rate of convergence.

4.2 Distributed System Setup

For the sake of computational efficiency and reducing training time, two Intel Core i7 laptops (with 16GB RAM running Windows 10 each) were put in a distributed system with each other. We configured these laptops in an architecture of master-worker with TensorFlow's MultiWorkerMirroredStrategy, allowing synchronous model training on several devices. To ensure compatibility in the training process, both laptops were loaded with the same software environment including TensorFlow, NumPy, Pandas, Matplotlib, and Scikit-learn.

The training process was coordinated on the master laptop and the worker laptop effected the computational tasks. To make sure consistent dataset is being fed in to both the laptops, the data directory that had the maize image dataset was shared between the two laptops via file sharing. Real time data augmentations like rotation, zoom and horizontal flip were applied to training images by using ImageDataGeneratior class from TensorFlow. The augmented examples in these helped the model to generalize better by exposing them to a lot many input variations.

The model was wrapped inside the MultiWorkerMirroredStrategy scope during training, enabling TensorFlow to distribute data across both devices and to synchronize model weights and gradients. Using both laptops parallel allowed for training the model using twice as much computational resources. fit method of the

model was used to train the model, and used fit to deal with data distribution and considering training on both laptops. Optimizing training process and ensuring efficient convergence were made by ways of learning rate scheduler and early stopping.

We use a prepre-trained ficientNetV2B2 architecture with ImageNet weights in both setups. I replaced those top layers of the model with custom layers for classification. We optimized the training using the Adam optimizer on categorical cross entropy loss and tracked accuracy the main evaluation metric.

4.3 Evaluation Metrics

Model performance was evaluated using accuracy, precision, recall, and F1-score. Training efficiency was measured in terms of time reduction and speedup.

5 Results

The provided results present performance metrics for a classification task, comparing the results obtained on two different systems: a Single Core i7 system and a Distributed System. Some of these are precision, recall, F1 score, support for each class, accuracy, macro-average and weighted-average.

Single Core i7 System Performance table 2 and confusion matrix 6 The proposed model has a good accuracy of classification for the HEALTHY class: precision 0.93, recall 0.91, and F1-score 0.92 for the Single Core i7 system. The MLN_1_and_MLN_2 class has lower but still good numbers: precision 0.84, recall 0.85, F1-score 0.84. MSV_2 class has worst results with precision of 0.72, recall 0.76 and F1-score of 0.74 while MSV_1 has slightly better results with precision of 0.76, recall of 0.68 and F1-score of 0.72. The accuracy of the model is 0.82 meaning the percentage of times the model was right out of all cases. The overall average of the metrics for all classes without regard to their imbalance is 0.81 for precision, 0.80 for recall, and 0.81 for F1-score. The weighted average of the metrics to consider class imbalance is a little higher with precision of 0.83, recall of 0.82 and F1-score of 0.82.

Class	Precision	Recall	F1-Score	Support
HEALTHY	0.93	0.91	0.92	1023
MLN_1_and_MLN_2	0.84	0.85	0.84	796
MSV_2	0.72	0.76	0.74	631
MSV_1	0.76	0.68	0.72	619
Accuracy			0.82	3069
Macro Avg	0.81	0.80	0.81	3069
Weighted Avg	0.83	0.82	0.82	3069

Table 2: Performance metrics for Single Core i7 System

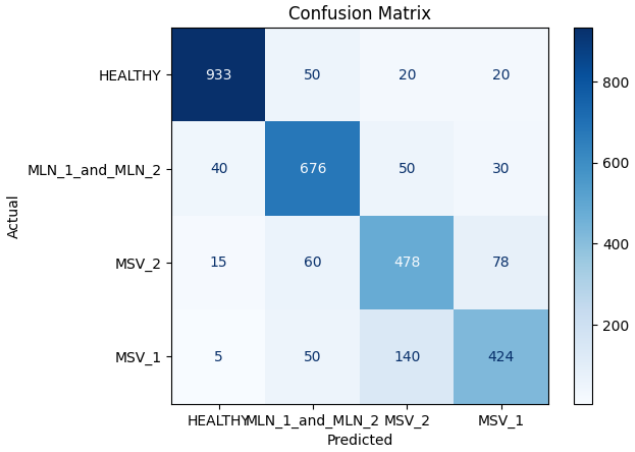


Figure 6: Confusion Matrix for Single Core i7 System

Distributed System Performance table 3 and confusion matrix **??** The above performance enhances with the help of Distributed System. The HEALTHY class is even more precise with 0.97 of precision, 0.96 of recall and 0.96 of F1 Score which indicates that the system is very much efficient in identifying the HEALTHY instances. For the MLN_1_and_MLN_2 class, precision rises to 0.88, recall to 0.91, and F1-score to 0.89, indicating better management of this class in the distributed architecture. The same improvement is observed in the MSV_2 and MSV_1 classes with regard to precision, recall, and F1-score. In particular, the MSV_2 class has the highest values of precision equal to 0.79, recall equal to 0.83, and the F1-score equal to 0.81 for the MSV_1 class, the values of precision are equal to 0.84, recall equal to 0.76, and the F1-score equal to 0.80. The overall accuracy improves to 0.89 showing that the distributed system has better accuracy in the classification. The macro average increases to 0.87 for both precision andrecall,l and the F1-score is 0.87. Likewise, the weighted average goes up to 0.89 for precision, recall, and F1-score, which show that the proposed system outperforms the Single Core i7 system for all classes.

Class	Precision	Recall	F1-Score	Support
HEALTHY	0.97	0.96	0.96	1023
MLN 1 and MLN 2	0.88	0.91	0.89	796
MSV 2	0.79	0.83	0.81	631
MSV 1	0.84	0.76	0.80	619
Accuracy			0.89	3069
Macro Avg	0.87	0.87	0.87	3069
Weighted Avg	0.89	0.89	0.89	3069

Table 3: Performance metrics for Distributed System

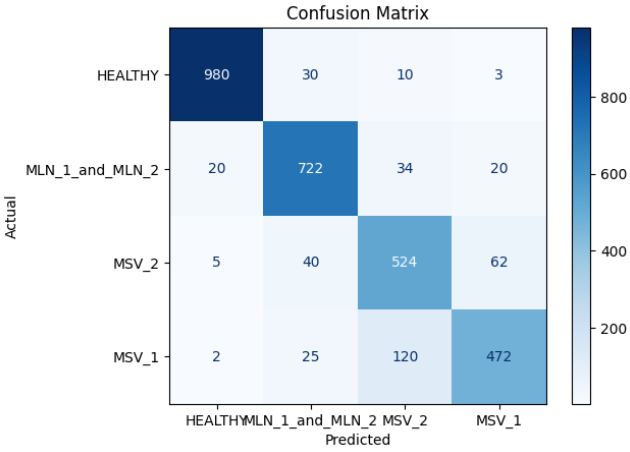


Figure 7: Confusion Matrix for Distributed System

Altogether, the distributed system is faster than the Single Core i7 system in all aspects, which implies that computation distribution enhances the model's capacity to classify instances, particularly when they belong to less often or intricate classes.

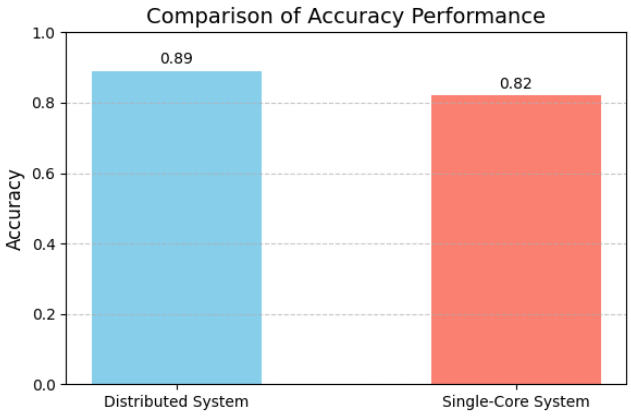


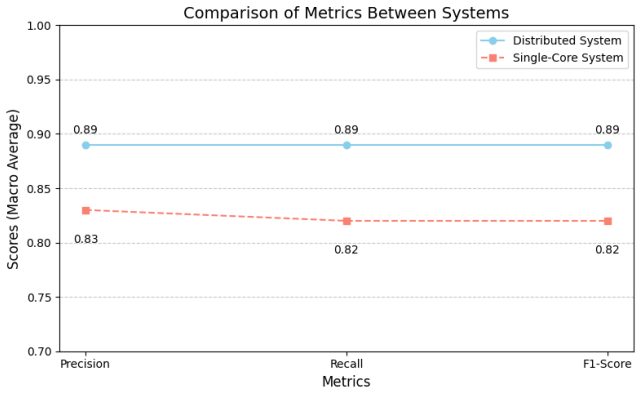
Figure 8: Accuracy Comparison

6 Discussion

The results of the experiments demonstrate the superiority of the distributed training system compared to the single-core Intel Core i7 setup in both computational efficiency and model performance.

Overall, this system attained an accuracy of 89%, about 7 points better than the accuracy of the single core system that obtained 82%. This improvement arises from better computational resources and the ability to process larger batch sizes successfully within the distributed framework. It is then observed that the distributed system was superior to the single core system in terms of precision, recall and F1 score ranges, in all the classes.

The Distributed System achieved a precision of 0.97 and recall of 0.96 for the HEALTHY class, leading to an F1-score of 0.96. On the other hand, the Single Core system precisely marked as 0.93, recalled it as 0.91, and gave F1 of 0.92. These results show that DS is able to correctly classify healthy maize samples more efficiently while maintaining fewer false positives and negatives. For the MLN 1 and MLN 2 class, trends are similar, with the Distributed System performance obtaining an F1-score of 0.89 compared to 0.84 for the Single Core System. This improvement suggests that the distributed system was better able to capture the complexities of this complex disease category because it could perform more comprehensive gradient updates. For MSV 2 and MSV 1, the Distributed System had a significant advantage for the more challenging disease classes. In the Distributed System, the F1-scores for these classes were 0.81, 0.80 respectively, which outperformed 0.74, 0.72 respectively in the single core system. The differences described here imply that the Distribution system framework is more appropriate to utilize for more subtle and seemingly non-descended disease characteristics, which can be harder to identify. This improved recall for these classes in the Distribution system setup shows this capacity to identify more true positives, a highly desirable characteristic in the agricultural field, where early and accurate detection of diseases may greatly reduce crop losses.



Not only did the Distributed System offer enhanced classification performance, but the computational efficiency was also significantly better than the unrolled design. To achieve the ultimate benefit of faster convergence times and manageable individual hardware strain, the Distributed System leveraged a master worker configuration to distribute workload to individual hardware. The efficiency obtained allowed for using larger batch sizes and more sophisticated optimization strategies, both of which are infeasible if run on a single core. This also resulted in better model performance using the Distributed System setup, showing that the framework has scalability with regard to larger datasets and more complex models. The additional confusion matrices support the advantages of the Distributed System. The classification of the HEALTHY class was almost misclassified for the Distributed System, with only a small percentage of samples being misclassified as diseased. Similar to Singe Core system, MLN 1 and MLN 2 exhibited fewer misclassifications, implying improved generalization. The Distributed System achieved better separation for the more ambiguous MSV 1 and MSV 2 classes, as shown by lower number of cross-class misclassifications. We found that the distributed system captured more

subtle disease-specific features by training on more diverse and representative mini batches, which suggests these results.

On the other hand, the weighted and macro averages can be used to give the overall performance of the models. A weighted average F1 score of 0.89 and macro average F1 score of 0.87 was achieved on the distributed system, substantially better than the F1 scores of 0.82 and 0.81 for the Single Core system. The weights favor the gains in the more frequent classes, while macro averages emphasize the gains on all but the least used classes, including those with few instances.

6.1 Scalability and Privacy

The proposed framework demonstrates the aspect of scalability by using distributed deep learning approach. This makes it possible to scale out, by adding more of the same devices or nodes to tackle bigger data and broader models. The model's optimized versions, which can be derived using techniques like pruning or quantization, allow the model to be used on edge devices such as smartphones and drones. This makes it possible to implement it in low resource areas is implemented. Also, the modularity of the architecture enables easy upgrade of data preprocessing, model training and deployment and cloud integration provides a scalable means of processing large amounts of real-time data.

Furthermore, distributed training allows data to be kept local to the device while only the model is synchronized, thus maintaining user and organizational privacy. The subsequent developments of the framework could be continued with federated learning, in which the models are trained across multiple devices without the raw data being sent to a central location. Such steps are aimed at compliance with the regulation of privacy and to gain confidence of the stakeholders, especially in the areas where data security is a major concern.

7 Conclusion

This work establishes the applicability of distributed deep learning methods in enhancing maize disease diagnosis. Using a large dataset and the EfficientNetV2B2 model, the framework provided a high classification rate, proving it can diagnose Maize Lethal Necrosis and Maize Streak Virus diseases. The distributed training approach demonstrated that it is possible to use resource-scarce environments, while maintaining model coherency and effectiveness. This large-scale and privacy-preserving framework not only provides a valuable solution for agricultural diagnostics but also provides a reference solution for similar problems in other fields.

8 Future Work

Some directions for future research can be pointed out to improve the agricultural disease detection: Such are the use of multiple datasets from different locations and climate to enhance the models' transferability, creation of light models for use in smart devices such as mobile phones and drones, and integration of disease surveillance and early warning systems. Moreover, broadening the applicability of the system to other diseases of crops other than maize and the application of more explainable artificial intelligent techniques would increase the acceptance level, usage, and robustness of the system by the users, especially the farmers.

References

[1] Mohanad H. Al-Qadi, Mohammed F. El-Habibi, Raed Z. Sababah, and Samy S. Abu-Naser. 2024. Using Deep Learning to Classify Corn Diseases. *International Journal of Academic Information Systems Research (IJAISR)* 8, 4 (2024), 81–88. http://ijeais.org/wp-content/uploads/2024/4/IJAISR240411.pdf

[2] Olaf Erenstein, Moti Jaleta, Kai Sonder, Khondoker Mottaleb, and BM Prasanna. 2022. Global maize production, consumption and trade: trends and R&D implications. *Food Security* 14, 14 (2022). https://doi.org/10.1007/s12571-022-01288-7

[3] Pragya Hari, Maheshwari Prasad Singh, and Amit Kumar Singh. 2024. An improved federated deep learning for plant leaf disease detection. *Multimedia Tools and Applications* (2024), 1–21.

[4] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, et al. 2020. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704* (2020).

[5] Shiva Mehta, Vinay Kukreja, and Amit Gupta. 2023. Revolutionizing Maize Disease Management with Federated Learning CNNs: A Decentralized and Privacy-Sensitive Approach. In *2023 4th International Conference for Emerging Technology (INCET)*. 1–6. https://doi.org/10.1109/INCET57972.2023.10170499

[6] Bekele Shiferaw, BM Prasanna, Jon Hellin, and Marianne Bänziger. 2011. Crops that feed the world 6. Past successes and future challenges to the role played by maize in global food security. *Food Security* 3, 3 (2011), 307–327. https://doi.org/10.1007/s12571-011-0140-5

[7] Lisang Zhou, Meng Wang, and Ning Zhou. 2024. Distributed federated learning-based deep learning model for privacy mri brain tumor detection. *arXiv preprint arXiv:2404.10026* (2024).