World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

DEEP LEARNING DRIVEN RECOGNISATIONAND CLASSIFICATION OF MAIZE LEAF DISEASES

Punnapu Suresh *, M. Sai Kumar, K. Nehal, S. Vasu and P. U. B. Jayalakshmi

Department of Electronics and Communication Engineering, Rajiv Gandhi University of Knowledge Technologies (RGUKT), Nuzvid.
* Corresponding Author

ORCID Details
Punnapu Suresh: https://orcid.org/0009-0008-0083-4490

World Journal of Advanced Research and Reviews, 2026, 32(02), 559–568

Publication history: Received on 28 September 2026; revised on 07 November 2026; accepted on 09 November 2026

Article DOI: https://doi.org/10.30574/wjarr.2026.32.2.2568

Copyright © 2026 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0

ABSTRACT

Leaf diseases of maize reduce yield and are usually found by walking the field and looking at the leaves, which is slow, depends on the observer and is hard to scale. Telling a healthy leaf from a diseased one is not enough, because the right response depends on which disease is present. This paper presents a four-class image classifier that labels a maize leaf as Healthy, Common Rust, Gray Leaf Spot or Northern Leaf Blight. The classifier is a pretrained YOLO11 nano classification model (yolo11n-cls) that was fine-tuned by transfer learning on the public Kaggle Maize Diseases dataset of 4,188 images. After exact duplicates were removed, the images were split 70/15/15 with stratification, and the under-represented Gray Leaf Spot class was balanced by oversampling with light augmentation applied to the training split only, so the validation and test sets contain original images alone. Training ran for 50 epochs at 224×224 resolution on a single Tesla T4 GPU and took about 18 minutes. On the held-out test set of 628 images the model reaches an accuracy of 97.13 percent, a macro F1 score of 0.9638 and a macro ROC-AUC of 0.9969. Healthy leaves are recognised without error, and Gray Leaf Spot is the weakest class with an F1 score of 0.9143, most often confused with Northern Leaf Blight. After layer fusion the model has 1.53 million parameters and its weight file is 3.2 MB. The work extends an earlier binary healthy-versus-diseased stage of the same project.

Keywords: Maize Leaf Disease, YOLO11, Transfer Learning, Multiclass Image Classification, Class Imbalance, Deep Learning

1 INTRODUCTION

Maize is a staple cereal crop, and its leaves are exposed to several foliar diseases that reduce the green area available for photosynthesis and, in severe cases, the final yield. Three of the most common are common rust, gray leaf spot and northern leaf blight. In most fields these are found by a person who walks the rows and looks at the leaves. This approach is slow, depends on the experience of the observer, and cannot cover large areas often enough to catch an outbreak early. An automatic classifier that works from a photograph of a leaf can support this scouting by giving a quick and consistent first opinion [1].

Deep convolutional networks are now the usual tool for this task. The YOLO family was introduced for real-time object detection [2], and the YOLO11 release also provides classification models that use the same efficient backbone [3]. Such a model can start from weights learnt on ImageNet [4] and needs only a short fine-tuning run on the target images, which suits a project with a small dataset and a single GPU [5].

This paper describes the design, training and evaluation of a four-class maize leaf disease classifier built on YOLO11. It extends the first stage of this project, which only separated healthy leaves from diseased ones, by naming the specific condition. The contributions of the work are listed below.

A four-class classifier for Healthy, Common Rust, Gray Leaf Spot and Northern Leaf Blight leaves, obtained by fine-tuning a pretrained yolo11n-cls model with a four-class output layer.

A data pipeline that guards against leakage: it removes exact duplicates, uses a stratified 70/15/15 split and applies oversampling with augmentation to the training split only.

Treatment of class imbalance, which raises the Gray Leaf Spot class from 402 to 914 training images while leaving the validation and test sets unchanged.

A full evaluation on a held-out test set with accuracy, macro and weighted precision, recall and F1, per-class scores, a confusion matrix and one-versus-rest ROC-AUC.

The remainder of the paper is organised as follows. Section II reviews the work on which this project builds. Section III explains the pipeline and Section IV the dataset and its preparation. Section V covers the model, the training set-up and the evaluation metrics, Section VI the results, and Section VII the limitations and the scope for future work. Section VIII concludes the paper.

2 BACKGROUND AND RELATED WORK

Image-based plant disease recognition has been studied widely with convolutional networks trained on photographs of leaves. Large public collections of labelled leaves have shown that such models can reach high accuracy when the images are taken under controlled conditions [1]. Two practical points limit how far these results carry. Public leaf images are often taken against plain backgrounds, and the number of images per disease is usually uneven, so a model can score well on a test split of the same collection and still struggle in the field.

The starting point of the present work is a study that describes a low-cost plant-scanner platform for detecting Ustilago maydis infection in maize with deep learning [6]. One of its approaches fine-tunes a pretrained YOLO11 model to separate infected from non-infected samples, which is a binary task on a private set of video frames. The present project keeps the central idea, a pretrained YOLO11 backbone adapted by transfer learning, but changes both the task and the data. It is not an attempt to reproduce the results of that study.

The first stage of this project used a public set of corn leaf images and answered the question of whether a leaf is diseased, with the two classes Healthy and Diseased (northern leaf blight). That output is of limited use, because the response in the field depends on which disease is present. The present stage answers the more useful question of which condition the leaf has, using a public four-class dataset [7]. Because the datasets and the number of classes differ, accuracy figures from the stages and from the reference study are not compared with each other.

3 SYSTEM ARCHITECTURE

The pipeline is organised in three blocks, shown in Table 1 together with the main elements of each. The data block collects the labelled images and prepares the splits. The model block holds the pretrained network and its training. The evaluation block measures the trained model on images it has not seen. Fig. 1 shows the order of the steps.

Table 1: Functional blocks of the classification pipeline

Block	Elements	Role
Data	Kaggle Maize Diseases images, corruption check, duplicate removal, stratified split, training-only oversampling	Produces clean training, validation and test sets with balanced training classes
Model	yolo11n-cls with ImageNet weights, four-class output layer, 224×224 input, 50-epoch fine-tuning	Learns to map a leaf image to one of the four classes
Evaluation	Best-epoch weights, accuracy, macro and weighted precision, recall and F1, confusion matrix, one-versus-rest ROC-AUC, single-image inference	Reports performance and gives a probability for every class

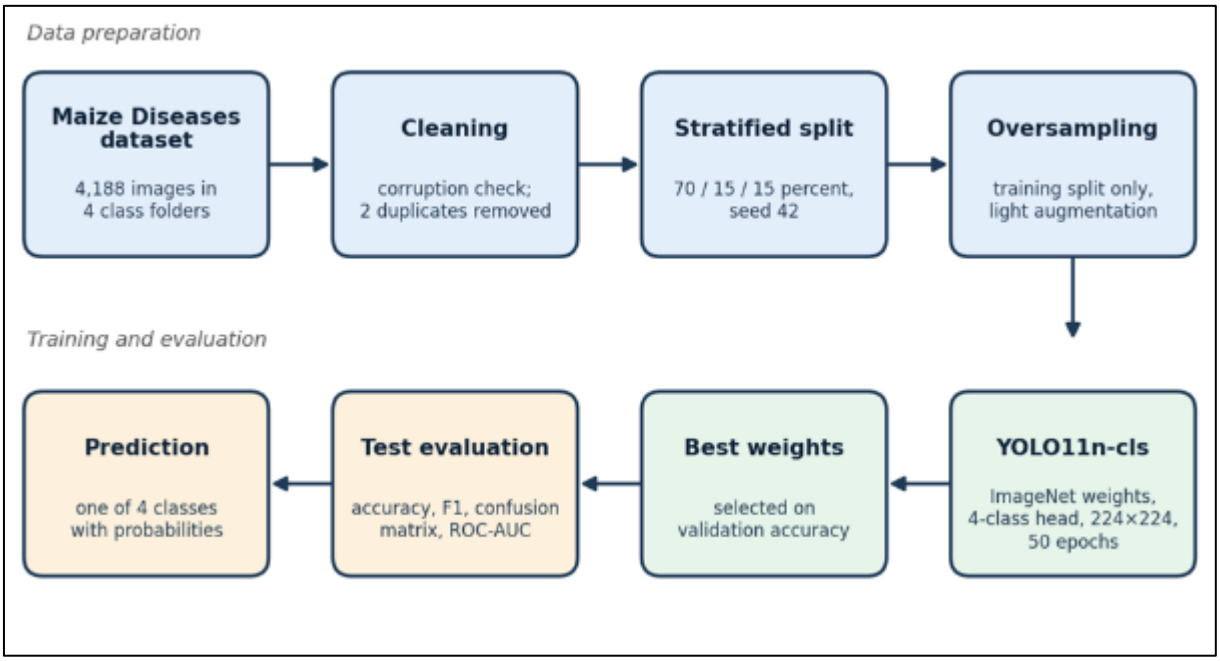


Figure 1: Pipeline of the maize leaf disease classification system

Class folders are detected by matching folder names against keyword lists instead of fixed paths, so the code works with different layouts of the same dataset. Folders that match no keyword are reported and skipped rather than ignored silently. The classifier outputs a probability for each of the four classes, and the class with the highest probability is the prediction.

4 DATASET AND PREPROCESSING

The images come from the public Kaggle Maize Diseases dataset [7], which is arranged as one folder per class: Healthy, Common Rust, Gray Leaf Spot and Northern Leaf Blight. The four folders hold 4,188 images. Common Rust is the largest class with 1,306 images (31.18 percent) and Gray Leaf Spot the smallest with 574 images (13.71 percent), so the ratio of the largest to the smallest class is 2.28. Fig. 2 shows the class distribution and Fig. 3 one image from each class.

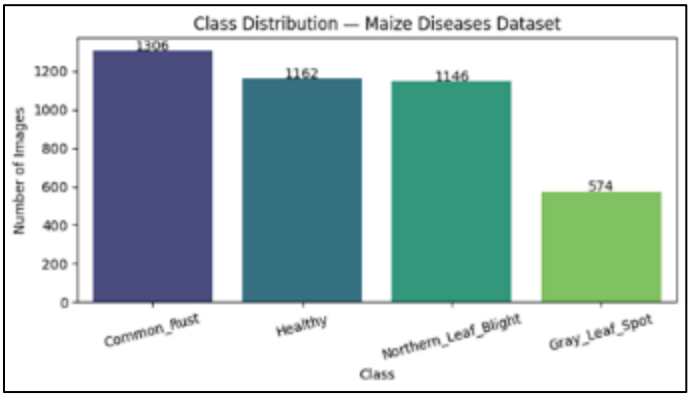


Figure 2: Class distribution of the Maize Diseases dataset



Figure 3: One representative image from each class

4.1 Cleaning and Splitting

No corrupted or unreadable image was found. A content hash identified two exact duplicate files, and both were removed before splitting to reduce the risk of one image appearing in two splits. The remaining 4,186 images were divided with a stratified 70/15/15 split and a fixed random seed of 42, so every split keeps the class proportions of the whole set. Table 2 lists the number of images in each split.

4.2 Handling Class Imbalance

Table 2: Images per class in each split

Class	After cleaning	Train (original)	Train (oversampled)	Validation	Test
Healthy	1,162	813	914	174	175
Common Rust	1,306	914	914	196	196
Gray Leaf Spot	574	402	914	86	86
Northern Leaf Blight	1,144	801	914	172	171
Total	**4,186**	**2,930**	**3,656**	**628**	**628**

Gray Leaf Spot has far fewer images than the other classes. Weighting the loss by class was not used, because the public training interface of the YOLO11 classifier does not expose a per-class weight argument. Copying images unchanged adds nothing new for the model to learn from. Oversampling of the minority class is a standard remedy for imbalance [8], and here extra copies were made with light augmentation: a random rotation of up to 15 degrees, a horizontal flip with probability 0.5, and brightness and contrast jitter of 0.15. The number of copies per class was limited to three times its original size and to the size of the largest class.

The oversampling was applied after the split and to the training split only. As Table 2 shows, every class then has 914 training images: 101 copies were added for Healthy, 512 for Gray Leaf Spot and 113 for Northern Leaf Blight, which gives 3,656 training images. The validation and test sets contain original images only, so the evaluation is not influenced by synthetic copies. Resizing to 224×224 is done by the YOLO11 training pipeline, and no other custom preprocessing is applied.

5 MODEL, TRAINING AND EVALUATION METRICS

The classifier is the nano classification model of the YOLO11 family, yolo11n-cls [3], initialised with weights pretrained on ImageNet [4]. As loaded it has 86 layers, 2,812,104 parameters and 3.3 GFLOPs; after layer fusion for inference it has 47 layers, 1,531,148 parameters and 3.2 GFLOPs. For fine-tuning, the trainer resizes the final layer to the four class folders found in the training directory. Table 3 lists the training configuration.

Table 3: Training configuration

Parameter	Value
Model	yolo11n-cls, pretrained on ImageNet
Input size	224×224
Batch size	32
Epochs	50
Initial learning rate	0.01
Random seed	42
Built-in YOLO augmentation	Disabled (colour, geometric, mosaic, mixup and erasing all off)
Hardware	Tesla T4 GPU (Google Colab)
Software	Python 3.13.15, PyTorch 2.11.0, Ultralytics 8.4.128
Training time	0.305 h (about 1,096 s) for 50 epochs

The random augmentation that the YOLO trainer normally applies was switched off. In this way the only augmentation present is the controlled oversampling described in Section IV, and two augmentation schemes are not mixed. The weights of the best validation epoch were kept for all evaluation. The network was trained with PyTorch [9].

5.1 Evaluation Metrics

Predictions on the validation and test sets were collected with the best weights, together with the probability of each of the four classes. All metrics were computed from these predictions with scikit-learn [10]. For class k, precision, recall and F1 score are given by Eq. 1 to Eq. 3, where TP, FP and FN are the true positives, false positives and false negatives of that class.

$$P(k) = TP(k) / (TP(k) + FP(k)) \quad (1)$$

$$R(k) = TP(k) / (TP(k) + FN(k)) \quad (2)$$

$$F1(k) = 2 \cdot P(k) \cdot R(k) / (P(k) + R(k)) \quad (3)$$

The macro average is the plain mean of the four class scores, for example macro $F1 = (1/4) \Sigma F1(k)$ for $k = 1$ to 4. It gives each class the same weight regardless of size. The weighted average weights each class by its number of images. The multiclass ROC-AUC uses a one-versus-rest scheme, in which each class in turn is treated as positive against the other three and the four areas are averaged [11].

6 RESULTS

6.1 Training Behaviour

Fig. 4 shows the loss and the validation accuracy over the 50 epochs. The training loss falls from about 0.50 in the first epoch to 0.0002 at the end. The validation accuracy rises quickly, shows a dip in the third epoch, and then settles between 0.95 and 0.96 with a best value of about 0.962; the last epoch gives 0.9586. The validation loss stays between roughly 0.14 and 0.31 and ends at 0.2533, which leaves a gap of 0.2531 from the training loss. This gap, and the slow upward drift of the validation loss after the first epochs, point to some overfitting, although the validation accuracy did not fall. The top-5 accuracy is 1.0 throughout and carries no information, since there are only four classes.

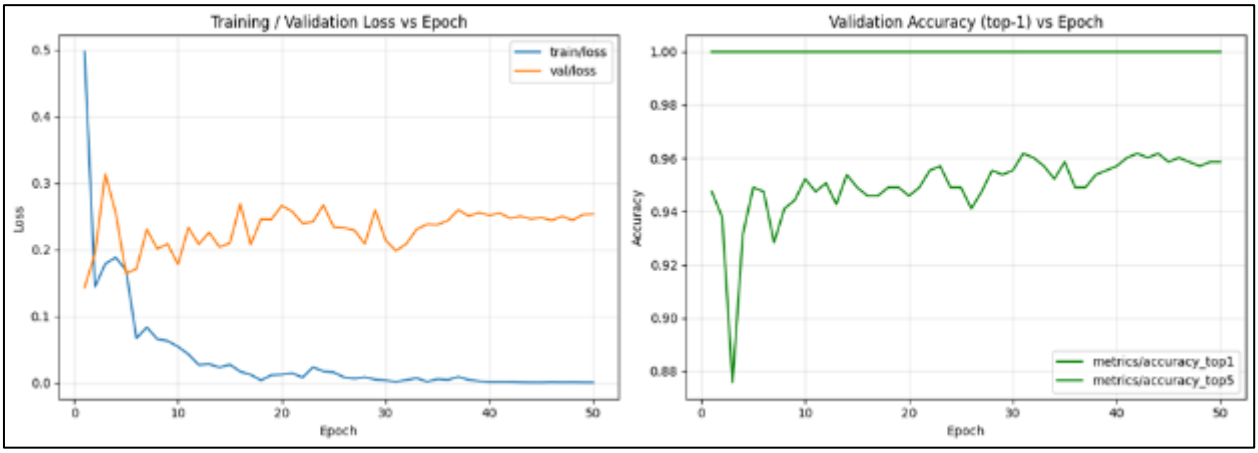


Figure 4: Training and validation loss and validation accuracy against epoch

6.2 Test Performance

Table 4 gives the overall scores on the validation and test sets, and Table 5 the scores for each class on the test set. On the test set the model reaches an accuracy of 97.13 percent, a macro F1 score of 0.9638, a weighted F1 score of 0.9714 and a macro ROC-AUC of 0.9969. The test scores are slightly higher than the validation scores (accuracy 96.18 percent). Both sets hold 628 images, so a difference of this size should not be read as more than a sign that the two results are of the same order.

Table 4: Overall results on the validation and test sets

Metric	Validation	Test
Accuracy	0.9618	0.9713
Macro precision	0.9533	0.9622
Macro recall	0.9482	0.9656
Macro F1	0.9505	0.9638
Weighted precision	0.9620	0.9717
Weighted recall	0.9618	0.9713
Weighted F1	0.9617	0.9714
Macro ROC-AUC	not computed	0.9969

Table 5: Per-class results on the test set

Class	Precision	Recall	F1 score	Support	ROC-AUC
Healthy	1.0000	1.0000	1.0000	175	1.0000
Common Rust	0.9797	0.9847	0.9822	196	0.9988
Gray Leaf Spot	0.8989	0.9302	0.9143	86	0.9923
Northern Leaf Blight	0.9701	0.9474	0.9586	171	0.9966

Figure 5 shows the confusion matrix of the test set as counts and as row percentages. The main observations are as follows.

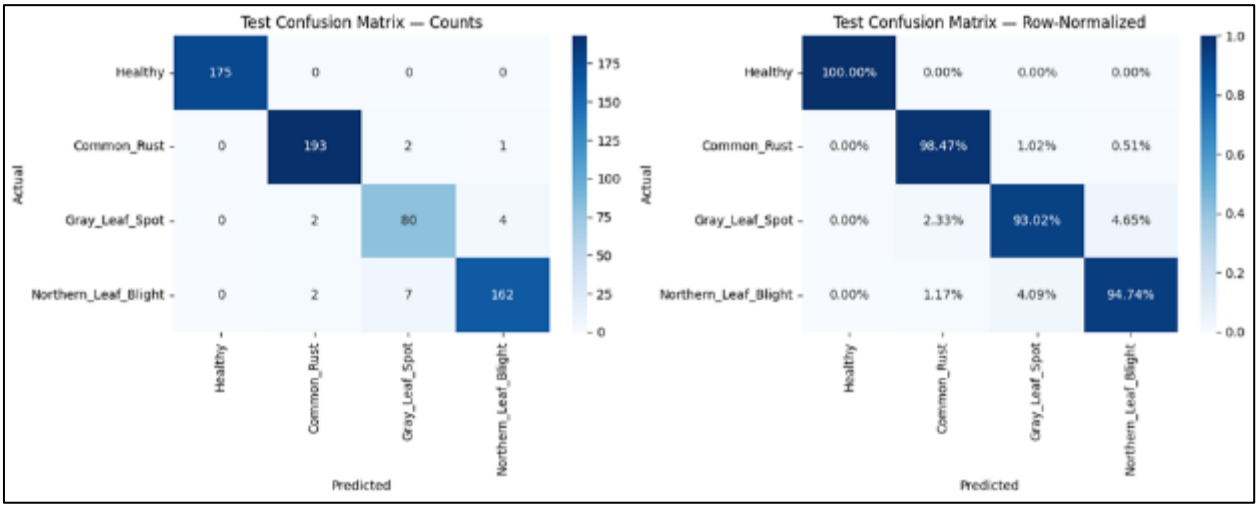


Figure 5: Confusion matrix of the test set (counts and row-normalised)

Healthy. All 175 healthy test leaves are classified correctly and no diseased leaf is called healthy, so precision, recall and F1 are all 1.0000.

Common Rust. 193 of 196 leaves are correct (98.47 percent); two are labelled Gray Leaf Spot and one Northern Leaf Blight. The F1 score is 0.9822.

Gray Leaf Spot. 80 of 86 leaves are correct (93.02 percent). The precision of 0.8989 is the lowest of the four classes and the F1 score of 0.9143 makes this the weakest class, even after oversampling.

Northern Leaf Blight. 162 of 171 leaves are correct (94.74 percent); seven are labelled Gray Leaf Spot and two Common Rust. The F1 score is 0.9586.

Main confusion. The most frequent error is Northern Leaf Blight predicted as Gray Leaf Spot (7 images). The opposite error occurs 4 times, so these two classes account for 11 of the 18 test errors.

ROC curves. The one-versus-rest AUC is at least 0.9923 for every class (Fig. 6), which shows that the class probabilities separate the classes well over a range of thresholds.

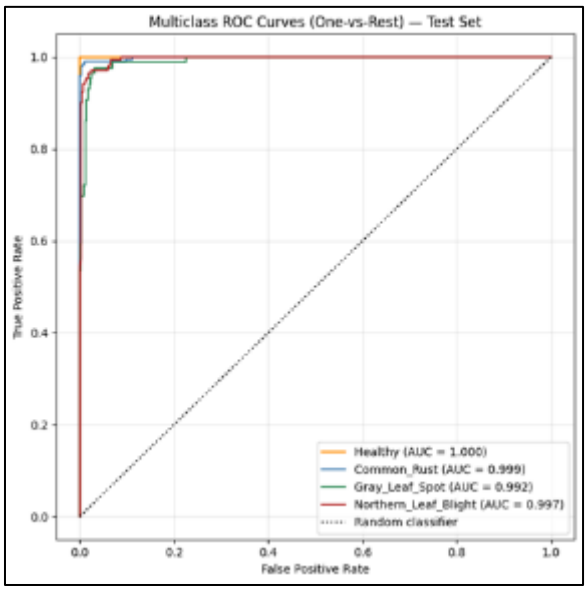


Figure 6: One-versus-rest ROC curves of the four classes on the test set

6.3 Sample Predictions

Fig. 7 shows six test images that the model classified correctly and Fig. 8 six that it classified wrongly, each with its actual class, predicted class and confidence. The images were drawn at random from the test set, so they illustrate the behaviour of the model and are not a representative count of errors. Every correct example in Fig. 7 is predicted with a confidence of 100.0 percent, and the images show leaves of different shapes and backgrounds.

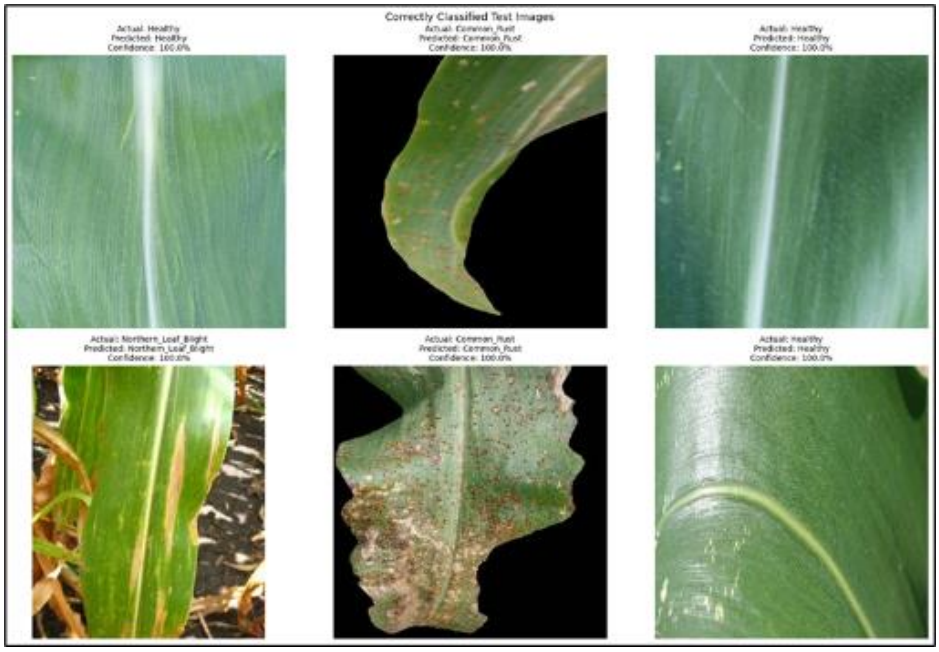


Figure 7: Correctly classified test images with actual class, predicted class and confidence

The errors in Fig. 8 differ in confidence. Three are made with a confidence between 48.9 and 65.1 percent, and such low values could be used to send a leaf for a second look. Two Northern Leaf Blight leaves, however, are labelled Gray Leaf Spot with a confidence of 99.9 and 99.5 percent, and one is labelled Common Rust with 82.1 percent, so the confidence value alone does not catch every error. The six images also vary in background, lighting and framing, with dark backgrounds, a leaf held in a hand, and close-ups of single lesions.



Figure 8: Incorrectly classified test images with actual class, predicted class and confidence

The trained model can also be applied to a single new photograph with a short inference function that prints the probability of each class. Fig. 9 shows the output for one test leaf, which is predicted as Healthy with a probability of 1.00, and the other three classes receive 0.00.



Figure 9: Single-image inference output of the trained classifier

6.4 Inference Speed and Model Size

In the validation pass on the T4 GPU the model needed about 1.0 ms per image for inference and 0.1 ms for preprocessing. The weight file of the best model is 3.2 MB. This small size and short inference time leave room for later use on modest hardware, although speed on a phone or an embedded board was not measured in this work.

7 DISCUSSION

Table 6 sets the present stage beside the reference study and the first stage of this project. The three differ in task, data and number of classes, so the table compares the problem definitions and not the accuracy values. A four-class problem is harder than a two-class one, and the aim of this stage is to give an output that can be acted on, namely the name of the disease.

Table 6: Comparison with the reference study and the first stage

Feature	Reference study	First stage	This work
Task	Binary infection detection	Binary disease detection	Four-class classification
Dataset	Private plant-scanner video frames	Public Corn Leaf Diseases (NLB)	Public Maize Diseases
Classes	Infected / non-infected	Healthy / Diseased (NLB)	Healthy / Common Rust / Gray Leaf Spot / Northern Leaf Blight
Question answered	Is the sample infected?	Is the leaf diseased?	Which condition does the leaf have?
Core method	Pretrained YOLO11, transfer learning	Same	Same

The results show that a very small YOLO11 model can separate the four classes of this dataset with errors that are concentrated in two of them. Oversampling brought Gray Leaf Spot to the same training size as the other classes, yet it still has the lowest F1 score. Synthetic copies add variation but no information that is absent from the original images, so they are no substitute for more real examples. A likely reason why Gray Leaf Spot and Northern Leaf Blight are confused with each other is that both produce lesions on the leaf blade, but the cause was not investigated here. No run without oversampling was made, so the size of its benefit is not measured in this paper.

LIMITATIONS AND FUTURE SCOPE

Several limits apply to the results. They come from one dataset and one seeded split, so they carry some variance that k-fold cross-validation would reveal. Only exact duplicates were removed, and near-duplicate images may still lie in different splits and make the scores optimistic. The lightweight yolo11n-cls backbone was chosen so that training fits

on a free Colab GPU, and larger YOLO11 variants were not tried. Most important, a high accuracy on a test split taken from the same collection does not show that the model works in the field; that needs validation on images taken under real conditions, with changing light, backgrounds and growth stages.

Several changes would move the work closer to use. A comparison with and without oversampling, and with class-weighted loss, would show which treatment of imbalance helps most. More real images of Gray Leaf Spot would address the weakest class directly. Larger model sizes and k-fold cross-validation would give a firmer estimate of performance. Bounding-box annotation would allow lesions to be located and several diseases on one leaf to be reported. Finally, packaging the model in a mobile application or an edge device, and testing it with agronomists in the field, would show whether the classifier is useful in practice

8 CONCLUSION

A four-class maize leaf disease classifier has been built by fine-tuning a pretrained YOLO11 nano classification model and evaluated on a held-out test set. The data pipeline removes duplicates, uses a stratified split and balances the training classes by oversampling with light augmentation, while the validation and test sets stay untouched. The model reaches a test accuracy of 97.13 percent, a macro F1 score of 0.9638 and a macro ROC-AUC of 0.9969. Healthy leaves are recognised without error. Gray Leaf Spot is the weakest class with an F1 score of 0.9143, and most errors lie between Gray Leaf Spot and Northern Leaf Blight. The model is small, with 1.53 million parameters after fusion, and fast, at about 1 ms per image on a T4 GPU. These results come from a single public dataset and do not by themselves establish effectiveness in the field, but they give a compact base on which field testing, larger models and deployment can be built.

STATEMENTS ON COMPLIANCE WITH ETHICAL STANDARDS

Acknowledgments

The authors thank the provider of the public Maize Diseases dataset on Kaggle, and acknowledge Google Colab for the GPU resources and the Ultralytics project for the open-source YOLO11 framework used in this work.

REFERENCES

[1] S. P. Mohanty, D. P. Hughes and M. Salathé, “Using deep learning for image-based plant disease detection,” Front. Plant Sci., vol. 7, art. 1419, 2016.

[2] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 779–788.

[3] G. Jocher and J. Qiu, “Ultralytics YOLO11,” Ultralytics, 2024. [Online]. Available: https://github.com/ultralytics/ultralytics

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2009, pp. 248–255.

[5] S. J. Pan and Q. Yang, “A survey on transfer learning,” IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[6] “A low-cost ‘plant-scanner’ platform for automated detection of Ustilago maydis infection in maize using deep learning.”

[7] S. Ghose, “Corn or maize leaf disease dataset,” Kaggle. [Online]. Available: https://www.kaggle.com/datasets/smaranjitghose/corn-or-maize-leaf-disease-dataset

[8] N. V. Chawla, K. W. Bowyer, L. O. Hall and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[9] A. Paszke et al., “PyTorch: An imperative style, high-performance deep learning library,” in Proc. Adv. Neural Inf. Process. Syst. (NeurIPS), 2019, pp. 8024–8035.

[10] F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[11] T. Fawcett, “An introduction to ROC analysis,” Pattern Recognit. Lett., vol. 27, no. 8, pp. 861–874, 2006.