INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 4, April 2026 | ISSN: 2456-4184 | IJNRD.ORG

AGROGUARD: AN AI-POWERED WEED DETECTION

[1]Mrs. A. Shalini, [2]Dr. A. Obulesu, [3]Pagidimarry Suhas, [4]Anthamgari Pramodhar, [5]Pilli Shiva Kumar,

[6]Ricabi Ujhwaal Sagar

[1]Assistant Professor, [2]Professor, [3,4,5,6]Student
[1,2,3,4,5,6]Information Technology
[1,2,3,4,5,6]Vidya Jyothi Institute of Technology, Hyderabad, India

Abstract: Weed infestation is a problem for farmers all over the world. It greatly reduces crop yield and overall productivity. Old ways of controlling weeds like removing them by hand and spraying chemicals everywhere take a lot of time are expensive and harm the environment. To solve these problems we created AgroGuard. It is a system that helps detect weeds. This system supports precise farming practices. AgroGuard uses the YOLO11 model to identify weeds from pictures of fields. This model is very good at finding objects in time. The system can recognize twelve types of weeds. It was trained on a varied set of weed pictures. We used techniques to make sure it works well in various field conditions, lighting and picture angles. Users can upload pictures of their fields. The system then accurately shows where the weeds are. Experimental results show that AgroGuard works with accuracy and reliability. It greatly reduces the time and effort needed to identify weeds. By allowing targeted weed management AgroGuard helps reduce chemical use. This contributes to friendly farming. This approach follows the principles of precision agriculture. Inputs are only applied where necessary. This leads to efficiency, reduced labor and optimized resource use. Overall AgroGuard is an effective solution for modern weed management challenges. It helps farmers deal with weed infestation in a way. AgroGuard makes farming more efficient and environmentally sustainable.

Index Terms - YOLO11, weed detection, precision agriculture, deep learning, object detection, smart farming, real-time detection.

I. INTRODUCTION

Sustaining food security for a growing population is a challenge in modern farming. One major issue is weed infestation. Weeds compete with crops for water, nutrients, sunlight and space. This competition can decrease crop yield by up to 34%. As a result, we have food shortages and financial losses for farmers worldwide. Traditional methods to control weeds include removal and chemical herbicides. However manual weeding is labor-intensive, time-consuming and costly. On the hand chemical herbicides can harm soil quality pollute water sources and lead to herbicide-resistant weeds. There is a need for a smarter approach to accurately identify weeds in field conditions. Computer vision provides a solution. By using learning on images from agricultural fields weed detection can be automated quickly and accurately[16]. This is a type of object detection that identifies objects in an image and determines their locations. The YOLO family of object detection models is known for its real-time detection with accuracy[11]. YOLO processes the image in one step predicting both the object type and its location. This makes it faster and more suitable for time agricultural applications. This work introduces AgroGuard, an end-to-end weed detection system built using the architecture. The system detects and locates twelve weed species from images taken in real field conditions. It has a FastAPI backend and a Streamlit web interface making it easy for farmers and experts to use without knowledge. AgroGuard helps reduce labor and excessive chemical usage supporting sustainable and precision agriculture goals with weed detection. AgroGuard uses weed detection to achieve these goals. The system relies on weed detection to help farmers. Weed detection is key, to AgroGuards functionality.

II. LITERATURE SURVEY

Research on finding weeds automatically has gotten a lot better over time. It used to be that simple picture editing methods were used. Now more advanced techniques like deep learning are used. In the systems people would manually pick out things like colour and texture and then use those with traditional classifiers [1]. With the introduction of convolutional neural networks this got a lot easier because these models can automatically figure out what is important in a picture and make detection more accurate [1].

The thing is, weeds and crops often look really similar which makes it hard to tell them apart. This is called the green-on-green problem [2]. For systems that need to work in time models like YOLO have become really popular because they are fast. A lot of studies have used YOLO models with types of crops and they found that when plants are all tangled up or the field is really crowded the model does not work as well [3]. When you compare YOLO models you can see that they can be really accurate but the quality and amount of training data is really important for how well the model works [4]. Another problem is when weeds are hidden behind crop leaves, which makes them hard to detect [5].

In the few years researchers have also been looking at using deep learning with pictures taken from the air. Some studies show that using intelligence to detect weeds and then spraying herbicides exactly where they are needed can use a lot less chemicals than traditional methods [6]. Others have used drone pictures with models to detect weeds really accurately but these methods often need expensive equipment [7]. There has also been work on using drones to monitor farm areas more efficiently [8]. To make these systems more useful smaller models have been made that can run on devices that're not very powerful but still give good results

[9]. However, with all this progress there is still a gap, between the models that researchers use and the systems that farmers can actually use which is what AgroGuard is trying to fix [10].

III. EXISTING SYSTEMS

Weed detection technology has gone through three phases, each defined by how computers process information. The first phase used created rules to identify weeds. These systems looked at areas that might be weeds and used things like color, texture and shape to decide if they were weeds. They used computers to make decisions but they weren't very good at handling changes in lighting, crop growth or soil. This approach was not very reliable. The second phase introduced a way of learning from images. Of using manually created rules computers learned to recognize weeds from labeled pictures [1]. This made the systems more robust and able to handle conditions. However early systems could only tell if a weed was present in a picture not where it was or what type of weed it was. The third phase improved on this by using a type of computer system that can detect and classify weeds in one step. Older systems, like Faster R-CNN were very accurate. Took too long to work in real-time [14]. Now systems like YOLO11 are the best using techniques and are also evolving significantly in successive versions[11][12], like anchor-free detection and attention-enhanced networks. These improvements help detect weeds and work more efficiently. Weed detection technology now uses single-stage object detectors to identify and classify weeds quickly and accurately. Weed detection systems have become more accurate and efficient over time handling environmental conditions. The use of CNN-based feature learning and single-stage object detectors has significantly improved weed detection.

IV. PROPOSED METHODOLOGY

AgroGuard is designed as a cohesive, modular pipeline that carries data from raw annotated imagery through a series of well-defined stages: preprocessing, augmentation, model training and optimization, performance evaluation, and web-based deployment. Each stage has been engineered to address a specific challenge encountered when applying deep learning to real-world agricultural imagery.

4.1 DATASET DESCRIPTION

The detection model is trained on 15,983 annotated field images sourced from an augmented version of the publicly available CottonWeedDet12 dataset [17][18], which covers twelve weed species common in cotton production systems. The pictures show twelve types of weeds that're bad for farming: Waterhemp, MorningGlory, Purslane, SpottedSpurge, Carpetweed, Ragweed, Eclipta, PricklySida, PalmerAmaranth, Sicklepod, Goosegrass and Cutleaf. All the pictures were taken outside in real conditions so they show how the light, soil and plants actually look. This is important because the system has to work in these conditions. The pictures are divided into groups for training, testing and validation. Tables 1 and 2 show how the pictures were divided and how many examples of each weed type are in each group.

TABLE 1: DATASET SPLIT DISTRIBUTION

Dataset Split	Image Count	Proportion (%)
Training	12,250	76.6
Validation	1,342	8.4
Testing	2,391	15.0
Total	15,983	100.0

TABLE 2: ANNOTATED INSTANCES PER WEED SPECIES

Species	Instances
Waterhemp	4,037
MorningGlory	2,678
Purslane	1,816
SpottedSpurge	1,632
Eclipta	1,704
Ragweed	1,692
Carpetweed	1,492
PricklySida	1,023
PalmerAmaranth	762
Sicklepod	527
Goosegrass	430
Cutleaf	257

4.2 DATA PREPROCESSING

Each image in the corpus goes through a process before it is used by the YOLO model. We make all images the same size, which is 640 x 640 pixels so they work with YOLO11. The images need to be this size to help the YOLO model work better. We change the brightness of the pixels to a range from 0 to 1. This helps the YOLO model work better and faster when it is being trained on YOLO11. We also check all the images to make sure they are good and not broken. I look at each image to see if it is okay. We get rid of any images that're bad or do not have the right information. The images must have this information. We put boxes around objects in the images in the YOLO format. This means we use the center of the box and how wide and tall it is. We use YOLO format for these boxes. We double check that every image has the information with it. This is done for all the images in the corpus. It makes sure everything is correct and in place, for YOLO11. The YOLO model uses these images to learn and get better at recognizing objects. The YOLO model learns from these images. These images help the YOLO model improve. The YOLO model gets better with YOLO11. YOLO11 helps the model learn.

4.3 Data Augmentation

The model needs to be good at working with all kinds of images not the ones it sees when it is learning. So, we use different ways to change the images online while the model is training[15]. We flip the images sideways sometimes. The model does not think that left or right matters. We also make an image by putting four different images together which helps the model see many different things in each group of images. This way the model can learn about different sizes and backgrounds. We mix two images together. Blend their labels, which helps the model learn softly. This means the model gets a push in the right direction. We take weeds from one image. Put them into another image, which helps because some weeds do not appear as much, as others. This helps the model learn about all the weeds. We change the colors of the images a bit by making them more or less bright and changing the colors. When the model is almost done learning we stop changing the images much so the model can settle down and learn the last few things. We do this for the part of the training time.

4.4 YOLO11 Architecture

The YOLO11 system works in a step figuring out the box around an object and what the object is at the same time[13]. It does this in one pass of doing it in two steps like some other systems. The main part of YOLO11 is built around something called C2f -stage partial blocks. These blocks help the system understand what it is looking at by using information from layers. This helps the system learn without getting too complicated. The neck part of YOLO11 combines information from the part. It takes information from the early layers and important information from the later layers. This helps the system understand where things are and what they are. The detection head is a part of YOLO11. It is separate. Does not use anchors. It has two jobs. Figuring out where the box is and what is inside the box. This helps the system learn faster and do a job. For this project the YOLO11n nano version was used. This version is small with about 2.6 million parameters. It is small enough to use in life but it is still good at detecting things. The YOLO11 system is a choice because it is small and good, at the same time.

4.5 Model Training

We use an NVIDIA Tesla T4 GPU to train our model. The Ultralytics training framework is what we use for this. We start with weights from a YOLO11n checkpoint that was pretrained on ImageNet. This really helps because the model does not have to learn everything from scratch. It already has an idea of what things look like so it can focus on learning about the agricultural domain. We train the model for a maximum of 100 epochs. Each batch has 16 images. They are 640 x 640 pixels. If the model is not getting better after 30 epochs we stop training. This prevents the model from getting too good at the training data and not good at anything. We save what the model has learned every 10 epochs. Then we pick the one to use. You can see all the details, about how we trained the model in Table 3. We trained the YOLO11n model. It is our model.

TABLE 3: TRAINING HYPERPARAMETER CONFIGURATION

Parameter	Assigned Value
Architecture	YOLO11n (~2.6M parameters)
Max Epochs	100 (early stop patience = 30)
Batch Size	16 images
Input Resolution	640 x 640 px
Hardware	NVIDIA Tesla T4 GPU
Initialisation	ImageNet Pre-trained
Augmentation	Mosaic, Mixup, Copy-Paste, HSV, Flip

4.6 Deployment Architecture

The trained model is part of a system that has three parts. This makes the model easy to use. It runs smoothly. The backend part is built with FastAPI. It has a part called an endpoint that takes pictures that users upload. This endpoint uses something called YOLO11 to look at each picture that is sent to it. Then it sends back a message with the things it found in the picture how sure it's that it found them and the picture with boxes around the things it found. It is good that the model is one instance. If we started an instance every time someone sent a picture it would take a long time to start especially if we are using the GPU. This would make it slow for people who want to know what is in their picture. The frontend part is made with Streamlit. This gives us a webpage where people can upload pictures from their computer or take a picture with their camera. The answers, including the picture with boxes and details are shown on the page. People do not need to know a lot of details to use the model. The model is designed to work in a way that does not make people wait it works with pictures. It gives clear answers. The model is easy to use. It gives people the information they want quickly. The trained model works well. It gives results when it is looking for species in pictures. The trained model is good at detecting species, in images. The trained model does what it is supposed to do.

V. RESULTS AND DISCUSSION

5.1 Evaluation Metrics

The performance of an object detection system is evaluated using standard metrics. One primary measure is Mean Average Precision at 0.50 or mAP50 for short which shows how well the model detects objects and classifies them correctly at a level. MAP50 is a starting point to see if a model can detect objects. A detailed metric is mAP50–95. It checks performance across levels of overlap. This gives an understanding of detection accuracy under stricter conditions. Precision is another metric. It is the number of detected objects divided by the total number of detections. This includes both incorrect detections. Precision shows how accurate the model's predictions are for object detection. Recall measures how many actual objects are detected. It shows how well the model finds all objects in an image. The F1 score combines precision and recall. It gives a view of overall object detection performance. The speed of detection is also important. It is measured by how images the model can process per second. This determines if the model is suitable for real-time applications. These metrics together provide an evaluation of an object detection system. They ensure that the system is both accurate and efficient, for object detection.

5.2 Quantitative Performance

Table 4 shows all the detection metrics for the three parts of the dataset. The difference in mAP50 between the training set and the test set is 1.67 percentage points. This is a sign. It means that using transfer learning adding types of augmentation and stopping all worked together to stop the AgroGuard system from getting too good at the training set and not good at the test set. We tested AgroGuard with 15,983 images. The test set mAP50 of 97.01 percent is really good. AgroGuard is better than systems that people have written about. The AgroGuard system can look at 83 pictures per second on the Tesla T4 GPU. This speed is fast enough to work in time. So AgroGuard can be used in the field. Work well with people. Looking at pictures quickly is important, for using AgroGuard in the field. The AgroGuard system can do this quickly.

TABLE 4: DETECTION PERFORMANCE ACROSS ALL DATASET PARTITIONS

Split	mAP50 (%)	mAP50-95 (%)	Precision (%)	Recall (%)	F1 (%)
Train	98.68	96.85	96.56	95.95	96.25
Validation	97.69	95.29	96.29	93.39	94.82
Test	97.01	94.21	95.28	92.67	93.96

5.3 Detection Visualization

Figure 1 shows what the model can do with images from the test set. These images have conditions like weeds close together different soil types and plants at various growth stages. The model draws boxes around the plants. Identifies their types correctly. It is also very confident in what it predicts. The model does well in scenes with many plant species together. It can tell apart plants that're close to each other even when their boxes overlap. This is usually hard for many systems to do. These results mean the model is good at finding and classifying plants in field conditions. Figure 1 clearly shows that the model works well in identifying plant species from field images. The model identifies plant species. It does this reliably. The results in Figure 1 are good. They are from real field images. The model is good, at its job.

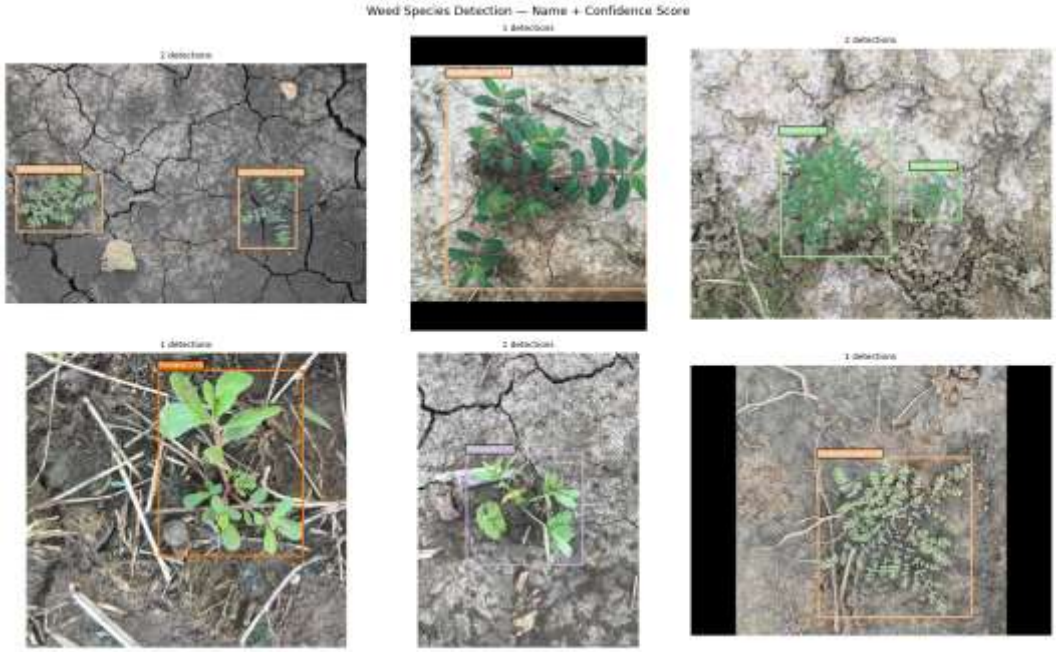

Figure 1: representative test-set inference outputs: bounding boxes, species labels, and confidence scores.

5.4 Confusion Matrix Analysis

Figure 2 shows us the confusion matrices for the test partition. We can see the count and the row normalized results. If we look at the diagonal, we can see that the model is really good at telling the twelve weed categories apart. The model has a time telling the difference between Carpetweed and Purslane. These two weeds look pretty similar to each other which makes it tough for the model to get it right. This is not a surprise because it is a problem. We have talked about this before. It is called the green on problem. If we add up the diagonal of the normalized matrix, we get a classification accuracy of 93.84, per cent for the twelve weed species. The model is able to classify the weeds most of the time.

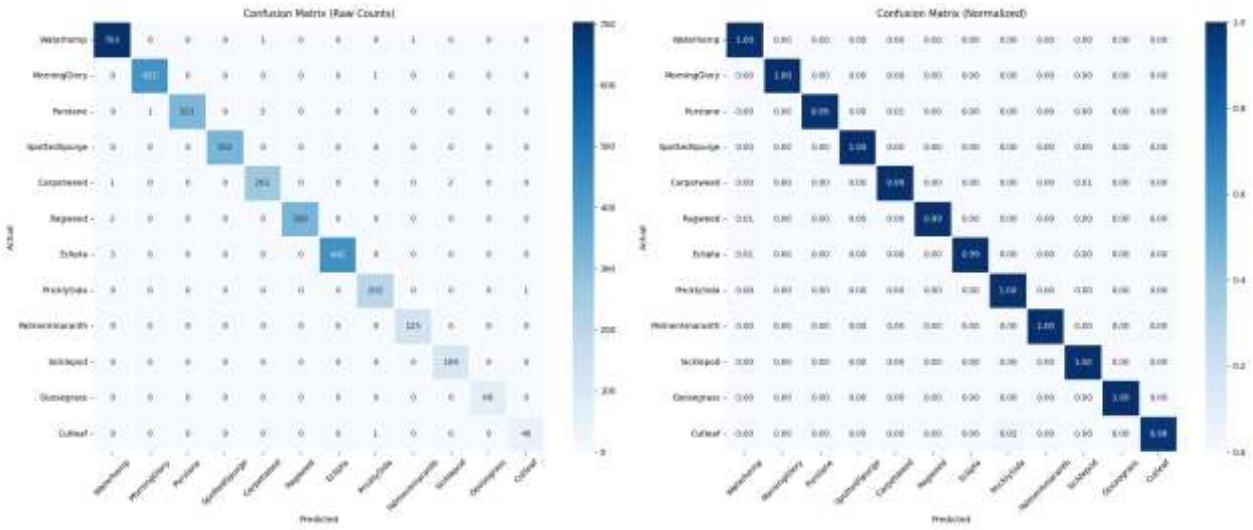

Figure 2: test-partition confusion matrix: absolute counts (left) and row-normalized proportions (right).

5.5 TRAINING CURVE ANALYSIS

Training progression over the 100-epochs schedule followed expected patterns, similar to what we see with transfer learning from a backbone pre-trained on ImageNet. The model learned fast with both mAP50 and mAP50-95 scores rising in the first twenty epochs. This rapid adaptation shows how well the pre-learned low-level feature detectors adjusted to the domain. After this fast phase both metrics levelled off at around 0.987 and 0.969 respectively and stayed stable for the rest of the training. The three loss signals. Bounding box regression loss, classification cross-entropy and Distribution Focal Loss. Each decreased smoothly and consistently without any signs of diverging. The validation loss tracked the training loss closely throughout with no gap between the two curves. This close tracking confirms that the model generalized well to images rather than just memorizing patterns from the training set. The model performed well on images it had not seen before which is a sign. The training progression was smooth and consistent, with no issues. The model seemed to learn and adapt well to the domain.

VI. CONCLUSION

This paper is about AgroGuard, which's a system that helps farmers find weeds. It uses YOLO11 technology and it is built to actually work on farms. We trained AgroGuard using a method, first we started with some -trained data, then we added data to it, and when the system was good enough we stopped training it. The results of AgroGuard are really good, the system was 97.01 percent accurate at finding weeds, it was 94.21 percent accurate at figuring out what type of weeds they were, and achieved a 93.96 percent F1 score for twelve different weed types. These results are better than what other similar systems achieve, so AgroGuard really helps improve weed detection technology. The proposed system can go through 83 images every second when it is running on a Tesla T4 GPU. It is also simple to use AgroGuard since you can get to it using a web browser. Future work will focus on further improvements to the proposed system, for example they want the system to work with video so that it could be used on vehicles and robotic sprayers, and they also want to shrink the model so it could run on devices like Raspberry Pi or even cell phones. The team also plans to add types of weeds to AgroGuards database which would help it work better in places, and are thinking about linking AgroGuard with GPS devices so that farmers can make maps showing how many weeds are in spots using AgroGuard.

REFERENCES

[1] K. Hu, Z. Wang, G. Coleman, A. Bender, T. Yao, S. Zeng, D. Song, A. Schumann, and M. Walsh, "Deep learning techniques for in-crop weed recognition in large-scale grain production systems: a review," ***Precision Agriculture***, vol. 25, no. 1, pp. 1–29, **2024**. DOI: 10.1007/s11119-023-10073-1

[2] A. S. M. Hasan, F. Sohel, D. Diepeveen, H. Laga, and M. F. Jones, "A survey of deep learning techniques for weed detection from images," Computers and Electronics in Agriculture, vol. 184, p. 106067, 2021.

[3] P. Ramesh, S. Kumar, and R. Raj, "Detection of weed location using YOLOv5 for real-time precision agriculture," International Journal of Advanced Research in Computer Science, vol. 16, no. 2, pp. 45-53, 2025.

[4] A. K. Kanade, R. Patil, and S. Deshmukh, "Weed detection in cotton farming using YOLOv5 and YOLOv8," Journal of Agricultural Engineering and Technology, vol. 33, no. 1, pp. 88-97, 2025.

[5] R. Goyal, S. Mehta, and A. Singh, "Deep learning-based weed detection in complex agricultural environments," Punjab Agricultural University Research Journal, vol. 12, no. 1, pp. 12-21, 2025.

[6] M. Guzel, H. Arslan, and T. Kaya, "Deep learning for image-based weed detection in wheat fields," Turkish Journal of Agriculture and Forestry, vol. 48, no. 3, pp. 210-221, 2024.

[7] H. Peng, Z. Li, W. Chen et al., "UAV-based intelligent weed detection using YOLOv8-PSPNet," Remote Sensing, vol. 15, no. 8, p. 2134, 2023.

[8] X. Huang, R. Liu, and Y. Zhang, "Deep learning for weed detection using UAV images," Computers and Electronics in Agriculture, vol. 196, p. 106954, 2022.

[9] J. Chen, M. Zhao, and S. Wu, "YOLO-CWD: A lightweight model for real-time crop and weed detection," IEEE Access, vol. 11, pp. 45312-45324, 2023.

[10] F. Abbas et al., "Deep learning-based weed detection in crops: A comprehensive review," Frontiers in Plant Science, vol. 13, p. 921580, 2022.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of IEEE CVPR, pp. 779-788, 2016.

[12] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv:2004.10934, 2020.

[13] Ultralytics, "YOLO11: State-of-the-Art Object Detection Model," 2024. [Online]. Available: https://docs.ultralytics.com/models/yolo11/

[14] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," IEEE Trans. PAMI, vol. 39, no. 6, pp. 1137-1149, 2017.

[15] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," Journal of Big Data, vol. 6, no. 1, p. 60, 2019.

[16] A. Kamilaris and F. X. Prenafeta-Boldu, "Deep learning in agriculture: A survey," Computers and Electronics in Agriculture, vol. 147, pp. 70-90, 2018.

[17] Y. Lu, "CottonWeedDet12: A 12-class weed dataset of cotton production systems for benchmarking AI models for weed detection," *Zenodo*, 2023. DOI: 10.5281/zenodo.7535814

[18] F. Dang, D. Chen, Y. Lu, and Z. Lewis, "YOLOWeeds: A novel benchmark of YOLO object detectors for multi-class weed detection in cotton production systems," *Computers and Electronics in Agriculture*, vol. 205, p. 107655, 2023.