AGRITHON 2.0

Organized by

School of Computer Science and Information Systems,

VIT, Vellore, India

Sponsored by

Department of Biotechnology, Govt. of India, New Delhi.

Team Members:

Name	Mobile Number	Email
Mohammed Arfath R	8667593576	mohammedarfath.r2022@vitstudent.ac.in
Naresh R	9043077326	naresh.r2022a@vitstudent.ac.in
Hariharan S	7418758902	hariharan.s2022d@vitstudent.ac.in
Mohammad Yusuf K A	8072750340	mohammadyusuf.ka2022@vitstudent.ac.in

Table of Contents

1. CVAT (Offline) Setup Procedure

To perform bounding box and segmentation annotations for crop images, we used **CVAT (Computer Vision Annotation Tool)** in its **offline Docker-based setup**.

1.1. Prerequisites

Before setting up CVAT locally, the following tools and system requirements were ensured:

- **Python** ≥ 3.8 (installed but not used directly for CVAT)
- **Docker Desktop** (with Docker Compose)
- **Git** (for cloning the repository)
- **Minimum 4–6 GB RAM free** for smooth container operations

1.2. Step-by-Step Setup

Step 1: Clone the CVAT Repository

```bash
git clone https://github.com/opencv/cvat.git
cd cvat
```

Step 2: Build and Launch CVAT using Docker

Inside the cvat/ directory, we executed the following:

```bash
docker-compose build
docker-compose up -d
```

⏳ The first-time setup might take 1–2 minutes. It pulls all dependencies and builds the containers.

Step 3: Access CVAT Dashboard

Once setup completed, we opened:

http://localhost:8080

Step 4: Account Creation

On the first launch, CVAT prompts for registration:

- We registered an **admin user account** to manage tasks and labels.
- All further annotations and uploads were done under this account.

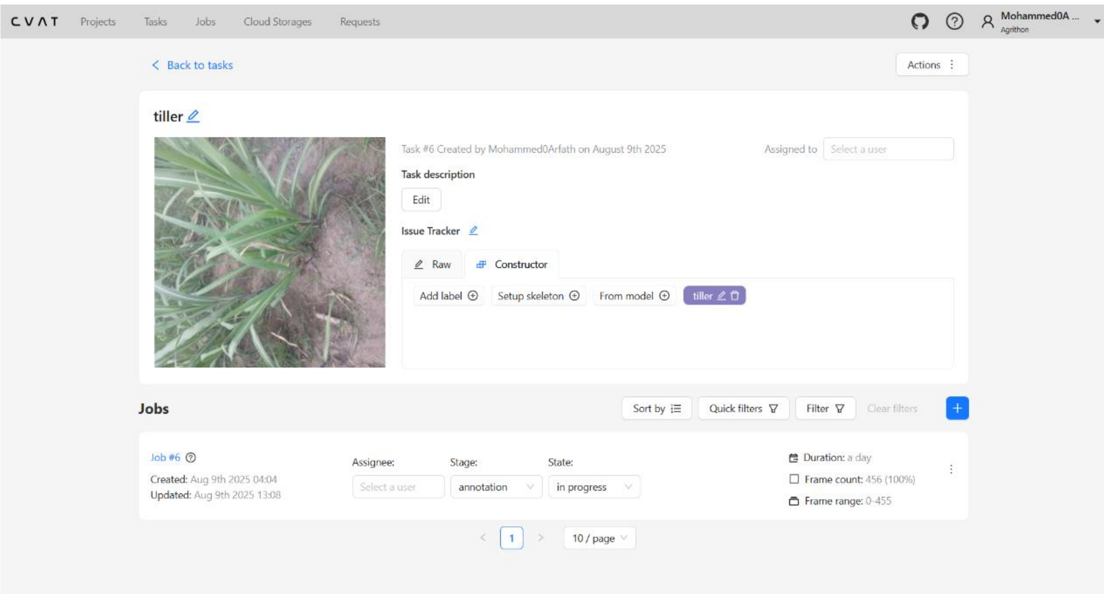
2. Annotation Techniques Used

Our annotation process was carefully tailored based on the nature of the two datasets: **Tillers** (requiring object detection) and **Dead heart** (requiring fine-grained segmentation).

2.1. Annotation Task Creation in CVAT

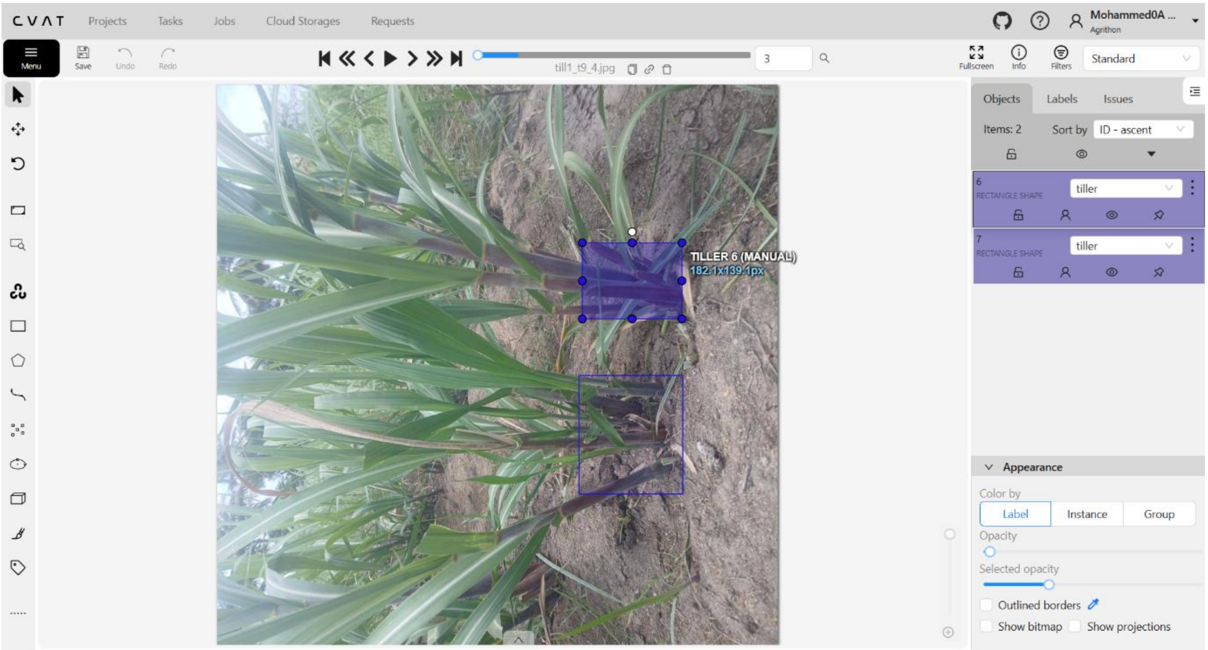
Here's how annotation was performed using CVAT:

- **Login** to CVAT a https://localhost:8080
- Click **"Create Task"**.
- Enter a descriptive task name:
    - Dead_Heart for segmentation
    - Tillers for detection
- Under **Labels**, click ➕ *Add Label*:
    - Use Dead_Heart for Dead Heart
    - Use Pillers for the Piller detection
- Choose Annotation Type:
    - Polygon for Dead_Heart (YOLOv8-seg) 🌿
    - Bounding Box for Tillers (YOLOv8-detect) 🐛
- Upload Images:
    - Use ZIP or drag-and-drop multiple .jpg/.png images
- Click **Submit**, then assign annotation jobs to team members.
- Start annotating through **"Jobs"** tab.



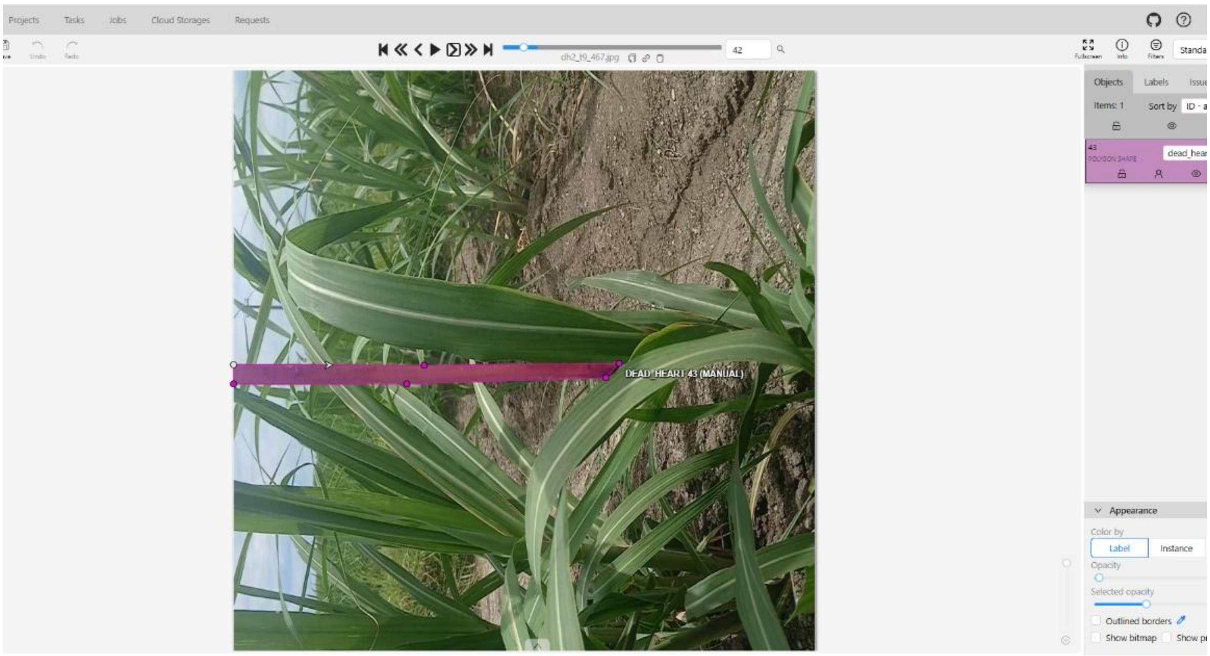
2.2.Tiller – Bounding Box Annotation

- **Tool Used:** Rectangle tool in CVAT
- **Class Label:** tiller
- **Annotation Type:** Object Detection (Bounding Boxes)
- **Objective:** Identify and locate tillers in sugarcane using YOLOv8-compatible box annotations.
- **Details:**
    - Annotated ~50 high-quality insect images manually.
    - Tillers was enclosed within a bounding box.
    - Special care was taken to avoid false positives like shadows or background noise.



2.3.Dead Heart – Segmentation Annotation

- **Tool Used:** Polygon tool + Brush mask in CVAT
- **Class Label:** dead_heart
- **Annotation Type:** Pixel-wise Instance Segmentation
- **Objective:** Accurately segment infected regions on leaves for disease localization and training with YOLOv8s-seg.
- **Details:**
    - Annotated ~50 leaf images manually with detailed masks.
    - Utilized brush tool for complex shapes like blight, mildew, and spot spread.
    - ⚠️ **Challenge Faced:** Manually brushing the mask for each image was time-consuming and required significant precision.
    - This process helped the model learn better edge localization and contour awareness.



2.4.Export Configuration (CVAT → Ultralytics)

- Tiller: **Ultralytics YOLO Detection 1.0**
- Dead Heart: **Ultralytics YOLO Segmentation 1.0**

These carefully annotated datasets laid the foundation for high-performance object detection and segmentation models using YOLOv8.

3. List of Tools & Packages Used

Core Libraries (Python Backend & AI Models)

Package	Version	Purpose
ultralytics	8.0.206	YOLOv8-based object detection & segmentation
torch	2.6.0	Deep learning framework for training & inference
torchvision	0.21.0	Computer vision utilities and pre-trained model support
pandas	2.1.3	Data manipulation and analysis
numpy	1.26.4	Numerical computations
opencv-python-headless	4.8.1.78	Image preprocessing and manipulation
scikit-learn	1.3.2	Machine learning utilities & evaluation metrics
pytorch-tabnet	4.1.0	TabNet model for questionnaire-based pest detection
albumentations	1.4.3	Data augmentation (via Roboflow pipeline)
fastapi	0.104.1	Backend web framework for REST API
uvicorn	0.24.0	ASGI server for FastAPI
python-multipart	0.0.6	File upload handling in FastAPI
aiofiles	23.2.1	Asynchronous file operations
Pillow	10.4.0	Image file handling (JPEG, PNG, etc.)

Frontend (User Interface)

Tool	Version	Purpose
React	18.x	Interactive web UI development
Vite	5.x	Fast frontend build and bundling tool
Node.js	20.x	JavaScript runtime for frontend development
CSS3	—	Styling and responsive design

DevOps & Deployment

Tool	Version	Purpose
Docker	26.x	Containerization of backend & frontend services
Docker Compose	2.x	Multi-container orchestration
Nginx	1.25.x	Reverse proxy and production web server

4. Architecture of the Multimodal System

4.1. Description

Our multimodal pipeline is specialized for detecting **two pest stages of the Early Shoot Borer (ESB)** in sugarcane:

- **Dead Heart Stage** (central whorl damage)
- **Tiller Stage** (multiple tillers from infested shoots)

The workflow begins with the annotation of raw sugarcane images in **CVAT (offline)** for each pest stage. Extensive augmentation is applied to boost model robustness under varying field conditions.

The processed datasets are then used to train the following specialized models:

- **YOLOv8s**: Trained on bounding box annotations for Dead Heart and Tiller detection (object detection).
- **YOLOv8s-seg**: Trained on segmentation masks to capture precise pest-affected regions.
- **TabNet Classifiers**: Trained separately on symptom-based textual inputs (Yes/No farmer responses) collected for each pest stage.

A **fusion module** aggregates predictions from both image-based and symptom-based models using a **confidence-weighted formula** to produce the final pest presence decision.

4.2. Pipeline Flow

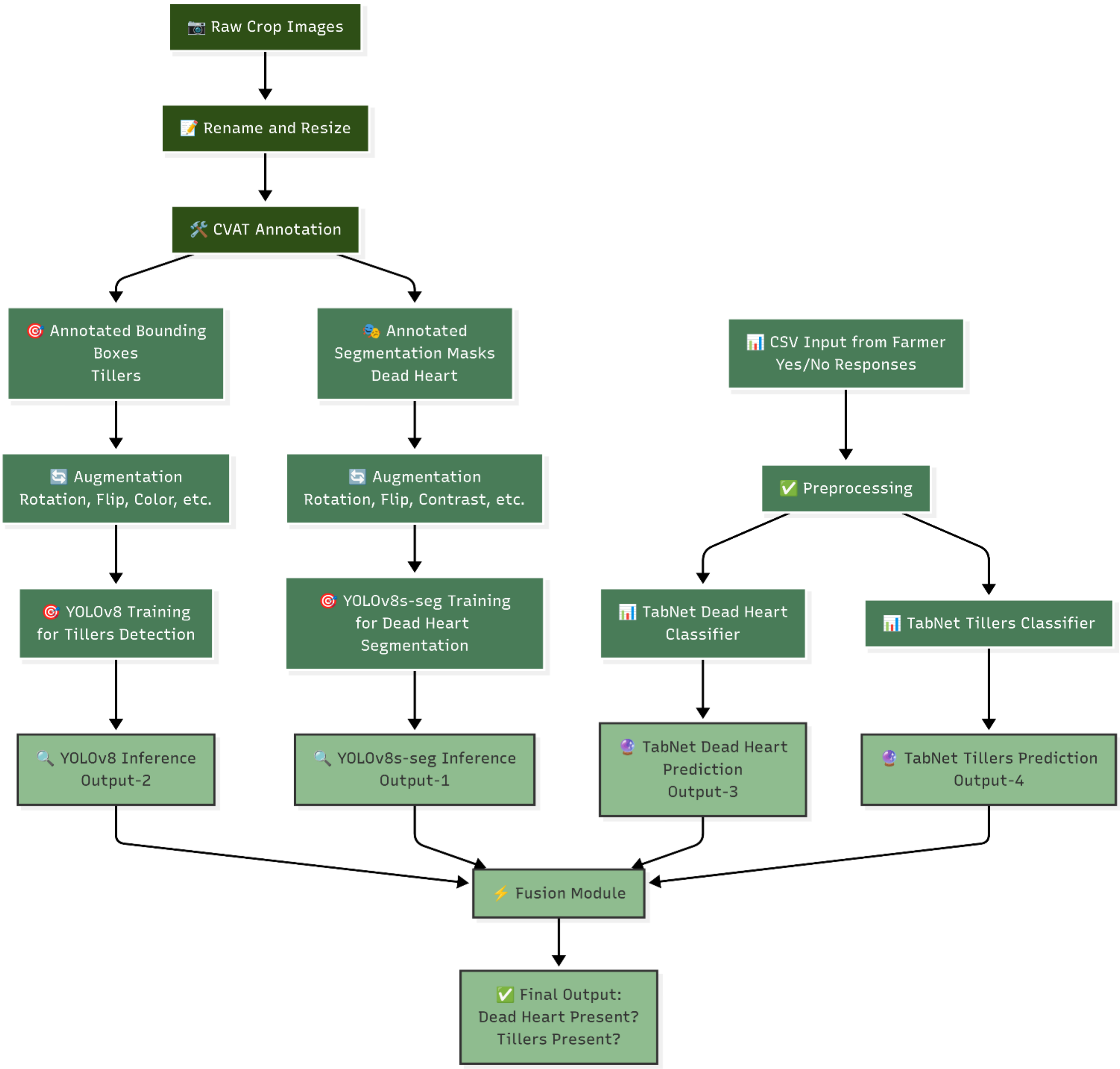
1. **Annotation Phase**
    - **Dead Heart**: Bounding boxes + segmentation masks
    - **Tiller**: Bounding boxes + segmentation masks
2. **Augmentation Phase**
    - Rotation, flipping, color jitter, brightness/contrast adjustment, normalization.
    - Stage-specific augmentation to mimic field variability (shadow effects, partial occlusion).
3. **Model Training Phase**
    - **YOLOv8s** (bounding box) and **YOLOv8s-seg** (segmentation) trained separately for each pest stage.
    - **TabNet** models trained on stage-specific symptom questionnaires (e.g., “Is the central whorl dry and easy to pull out?”).

4. **Inference Phase**
    - Image inputs processed by YOLO models.
    - Farmer symptom inputs processed by TabNet models.
5. **Fusion & Decision Phase**
    - For each pest stage, results from image and symptom models are combined using:

Final Confidence = (0.6 × YOLO Confidence) + (0.4 × TabNet Confidence)

- Final outputs:
    - **Dead Heart Present?** ✅ / ❌
    - **Tiller Present?** ✅ / ❌

4.3. Architecture Diagram



4.4.YOLO Model Overview

YOLOv8s Architecture Breakdown:

Component	Description
Backbone	C2f blocks with Conv + BN + SiLU activations. It uses a variation of CSPNet to reduce computation while preserving accuracy.
SPPF (Spatial Pyramid Pooling - Fast)	Enhances receptive field with max-pooling layers in parallel.
Neck	PANet (Path Aggregation Network) for feature fusion from different scales.
Head	- **YOLOv8s**: 3-scale detect head - **YOLOv8s-seg**: includes additional **segmentation head** on top of detection.
Activation	Uses **SiLU (Swish)** activation throughout for better nonlinear performance.
Loss Functions	- **CIoU Loss** for bounding boxes - **BCE Loss** for classification & segmentation masks.

Number of Parameters:

Variant	Parameters	Speed	Use Case
YOLOv8s	~11.2M	⚡ Fastest	Real-time detection
YOLOv8s-seg	~12.5M	⚡ Slightly slower	Real-time + segmentation

Activation Functions:

Layer Type	Activation Used
Convolutional	SiLU (Sigmoid-weighted Linear Unit)
Detection Head	Linear outputs (for regression)
Segmentation Head	Sigmoid + BCE loss

Training Optimizer:

- Optimizer: **AdamW**
- LR scheduler: **Cosine Decay**
- Mixed precision (fp16) enabled for faster training.

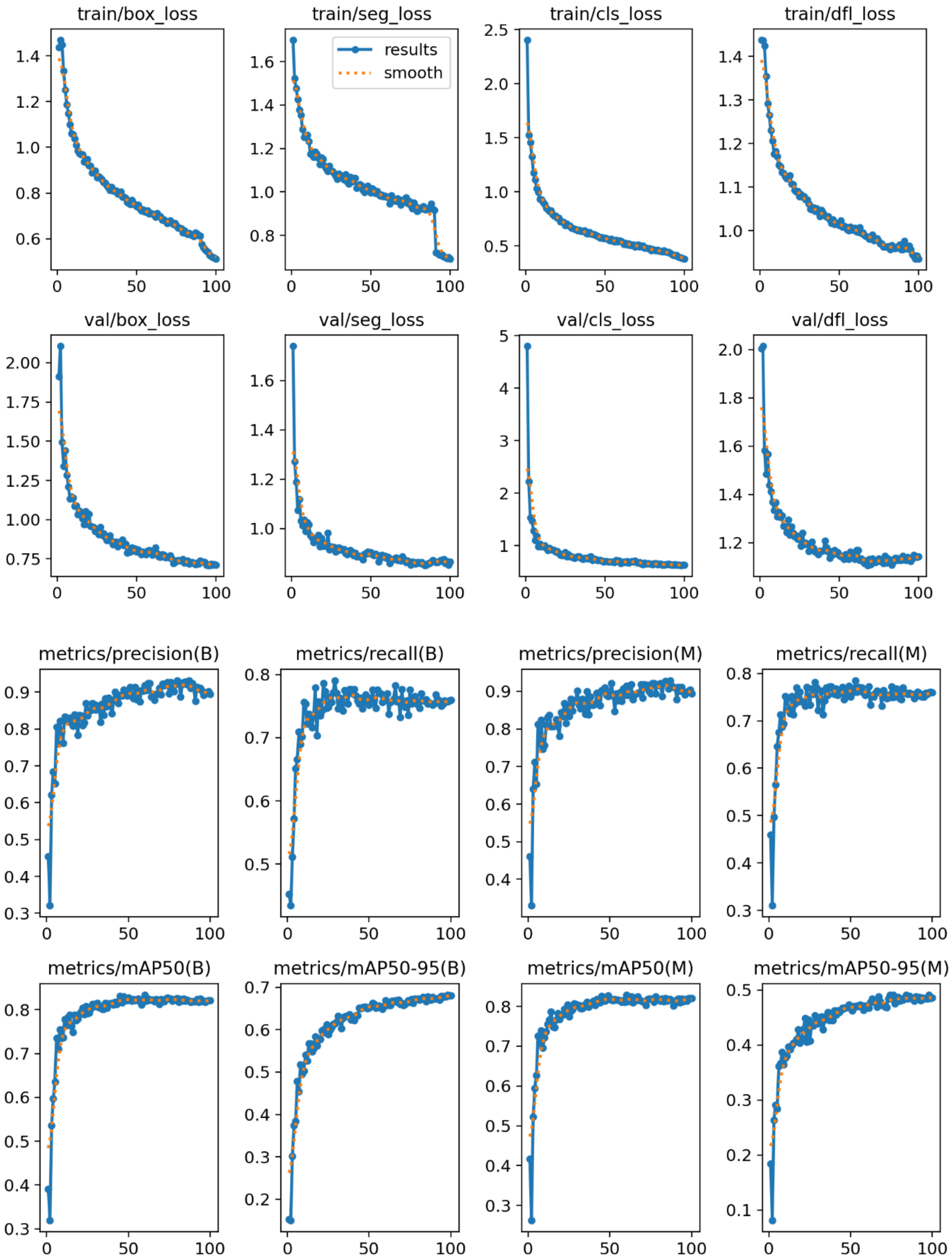
5. Output Screenshots & Performance Highlights

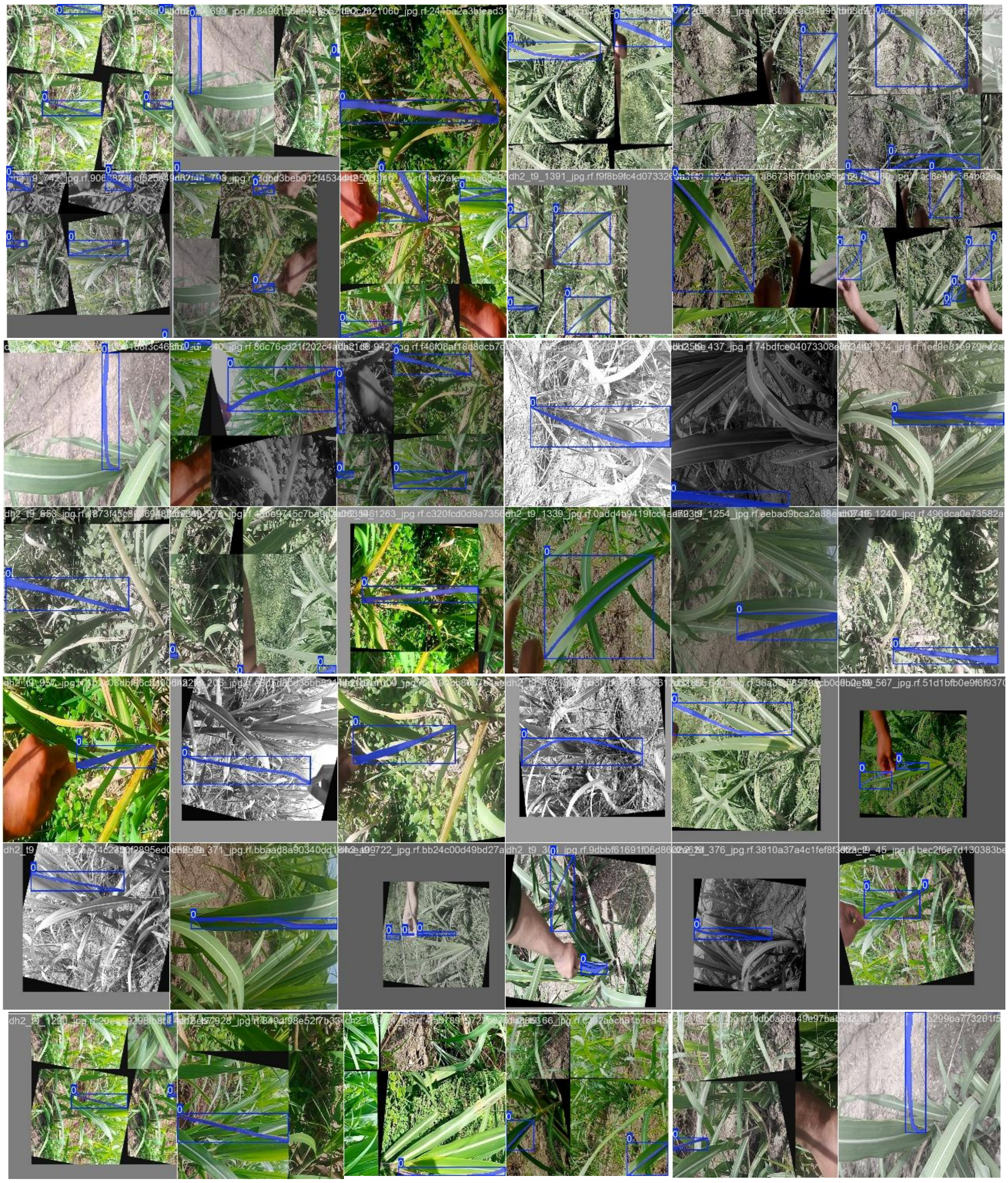
5.1. YOLOv8s Tiller Detection

5.2. YOLOv8s-seg Disease Segmentation

6. Performance Metrics

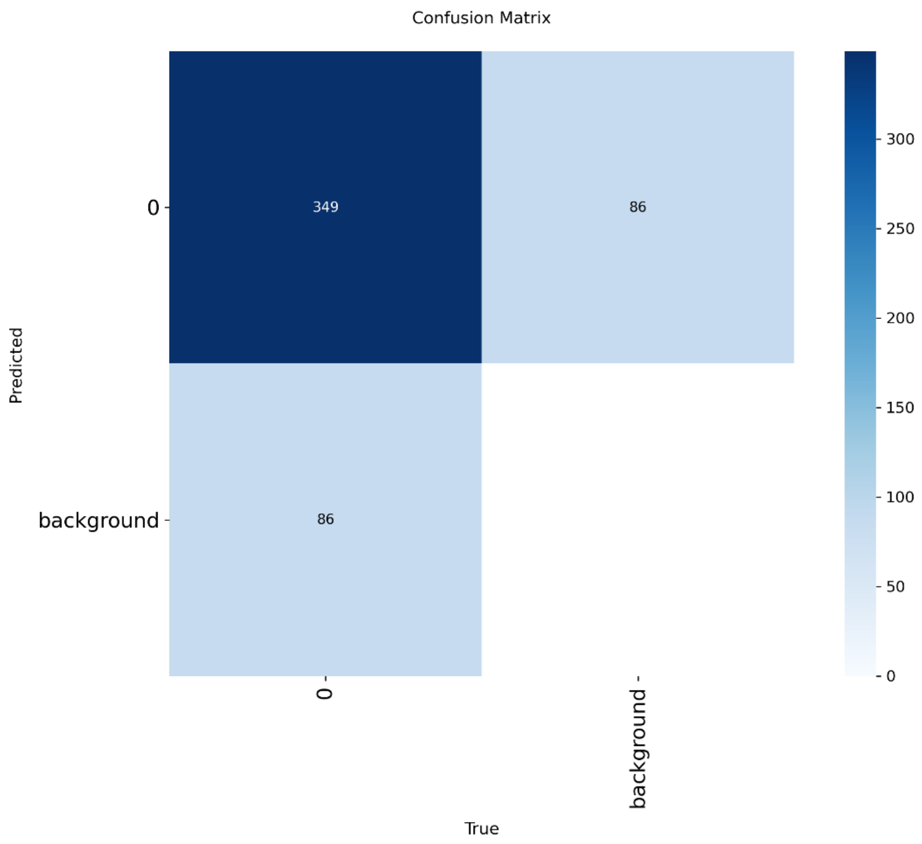
6.1. YOLOv8s-seg Dead Heart Segmentation Model Metrics



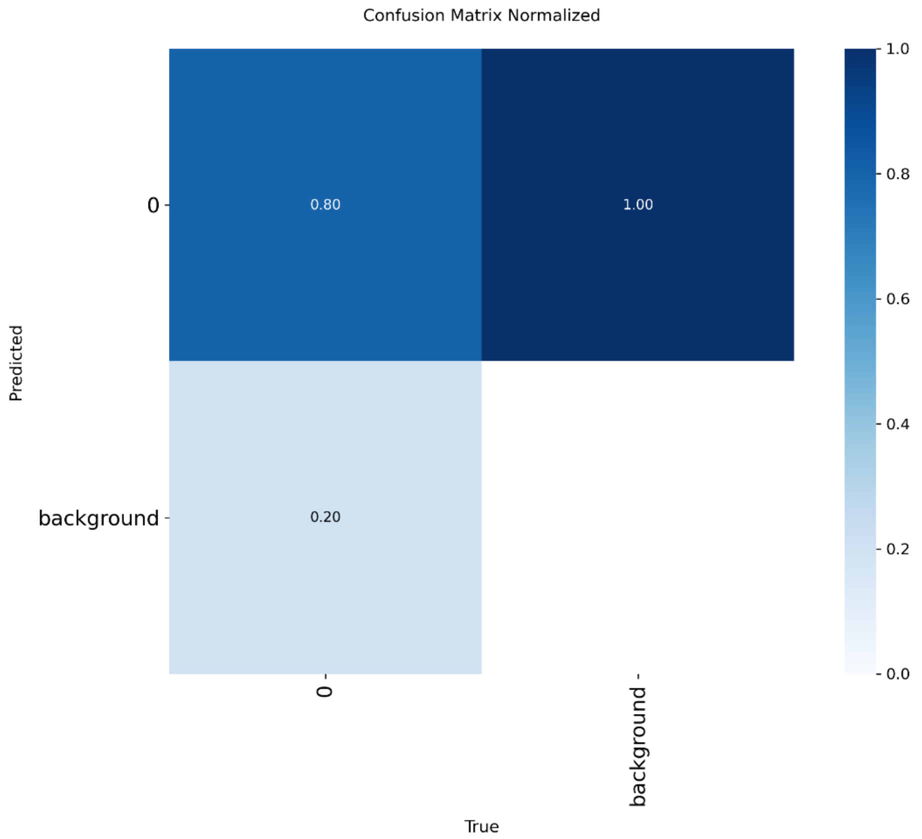


Training

Validation

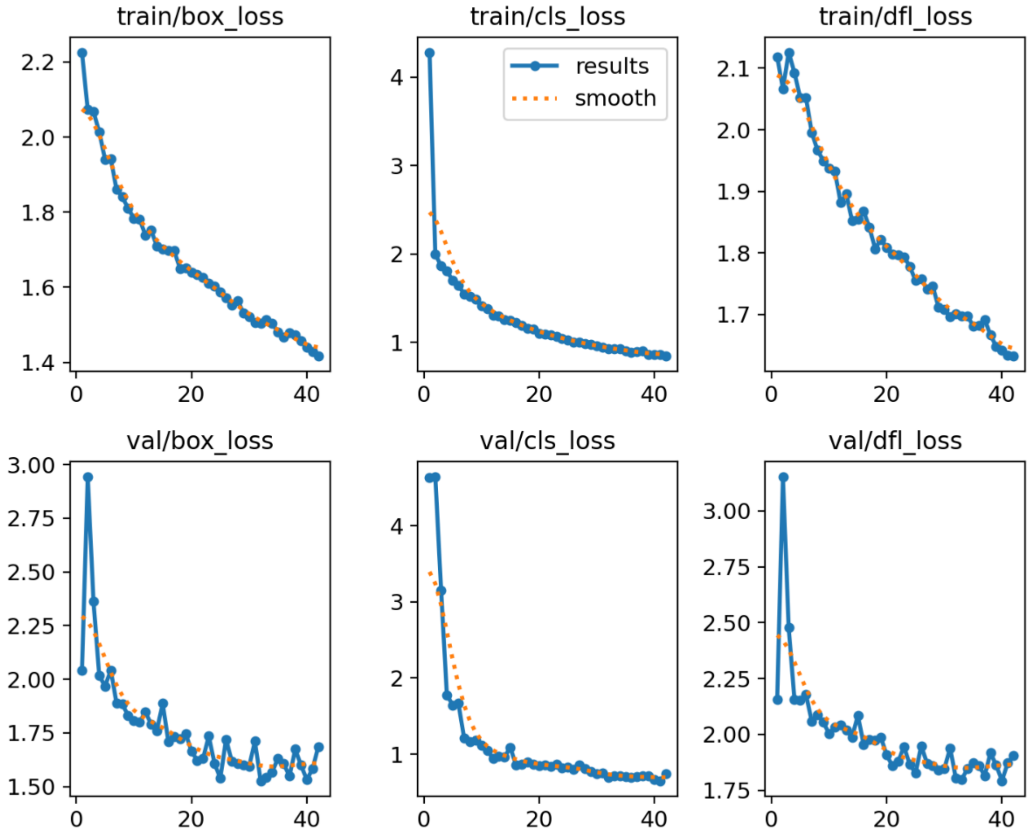


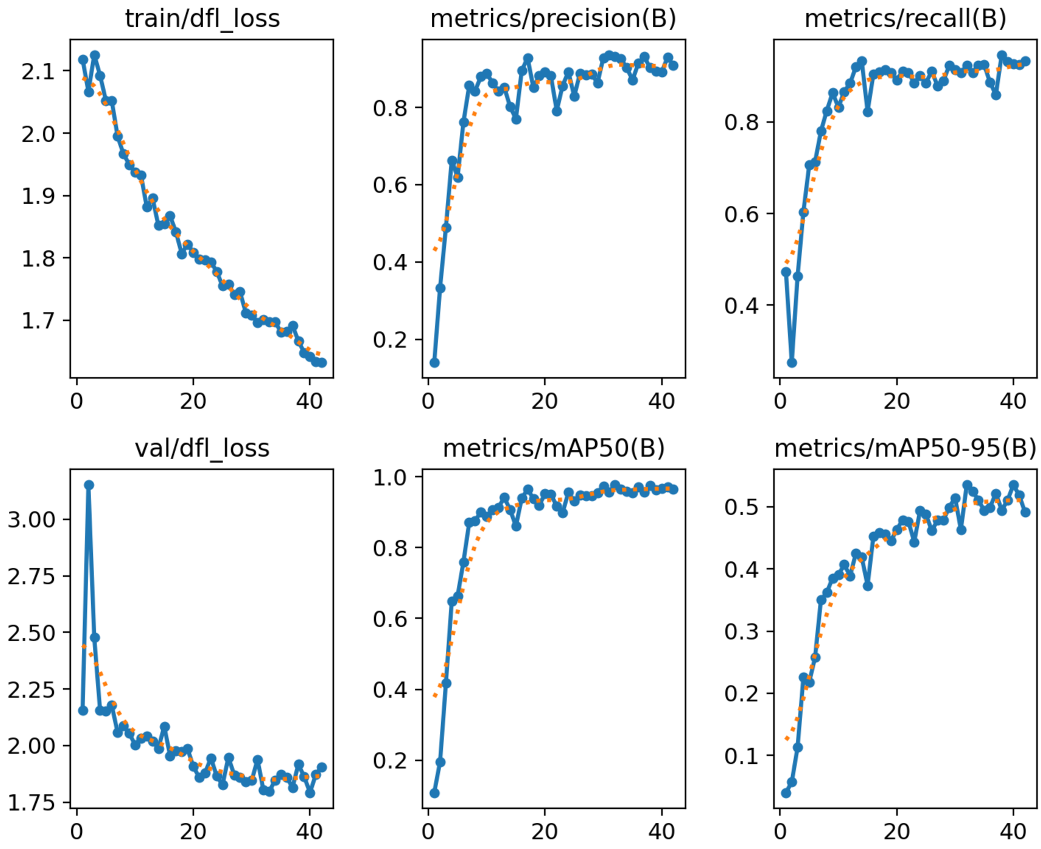
Confusion Matrix
Predicted
0
background
349
86
86
0
background
True
300
250
200
150
100
50
0

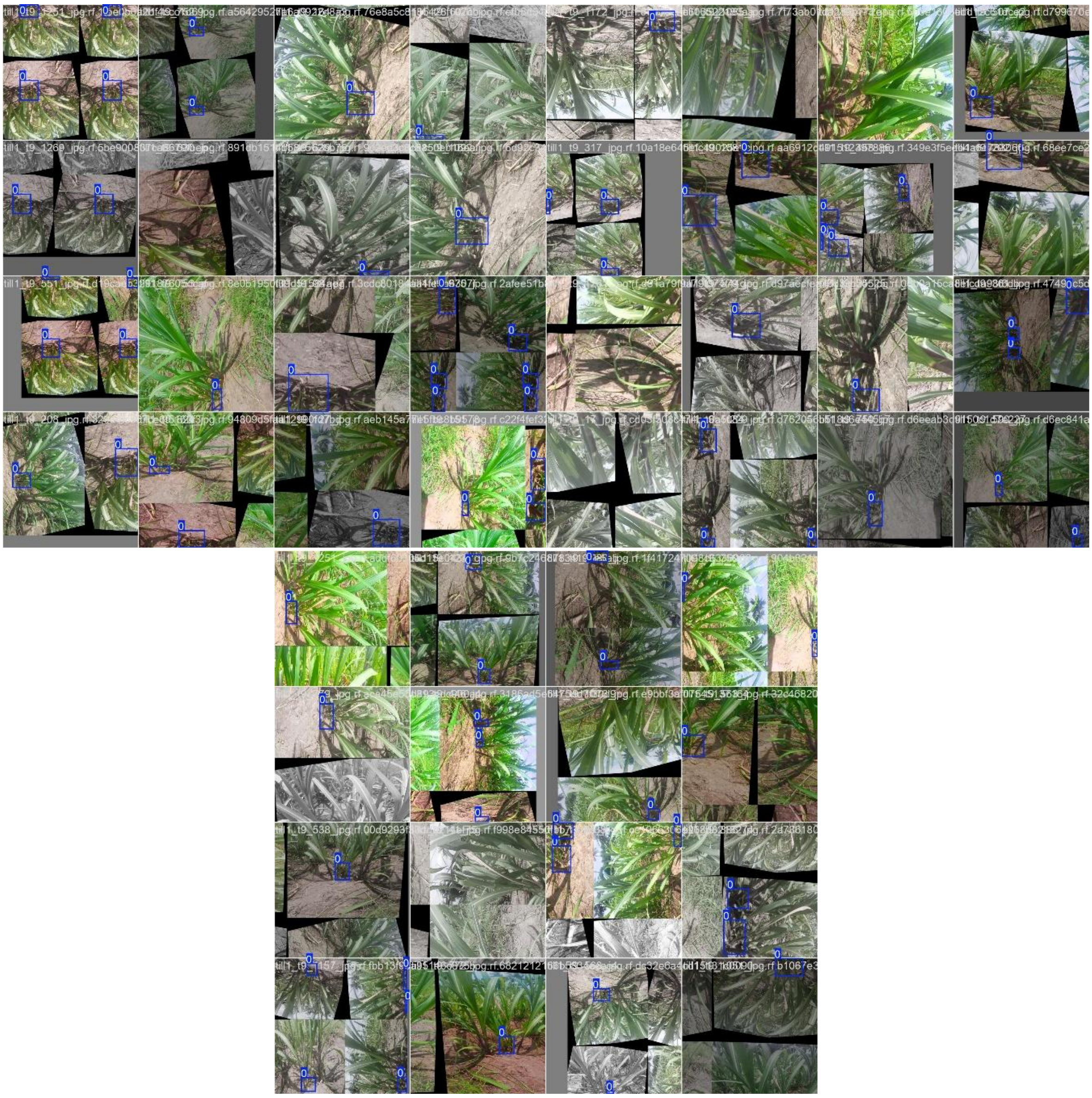


Confusion Matrix Normalized
Predicted
0
background
0.80
1.00
0.20
0
background
True
1.0
0.8
0.6
0.4
0.2
0.0

6.2.YOLOv8s Tiller Detection Model Training Metrics

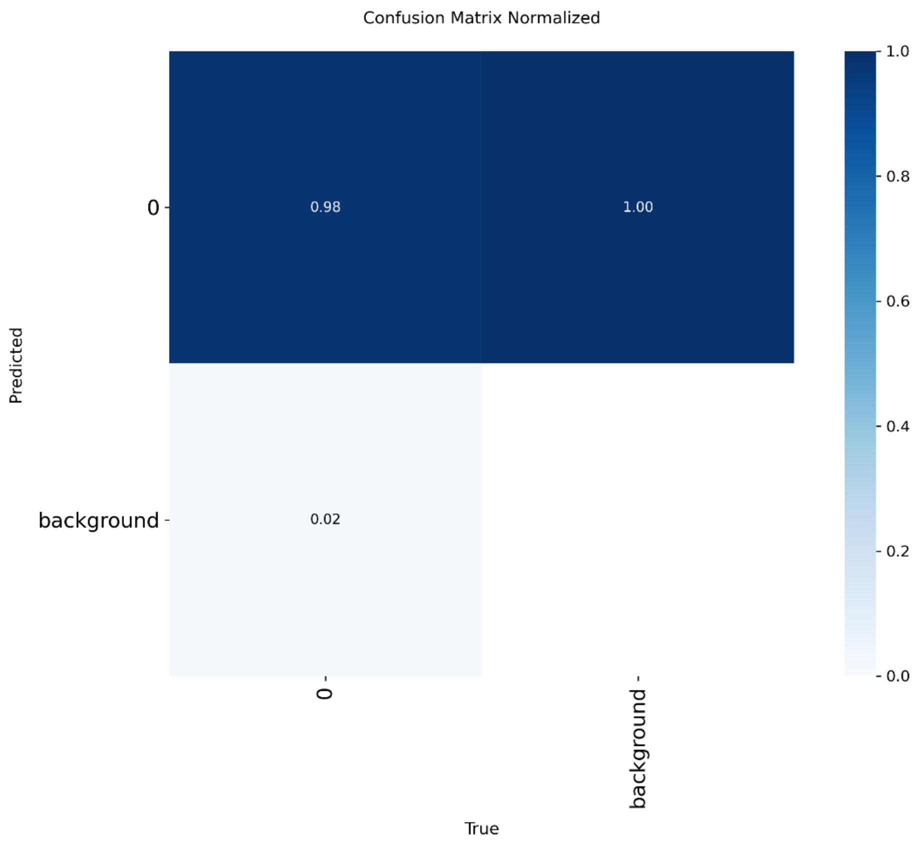


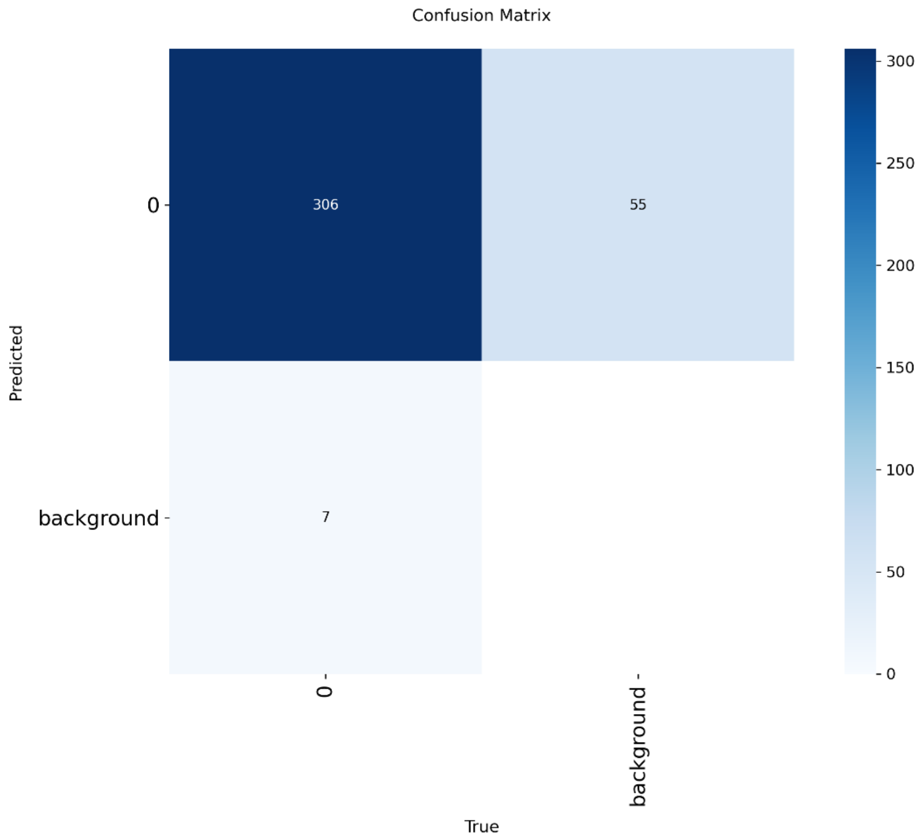




Training

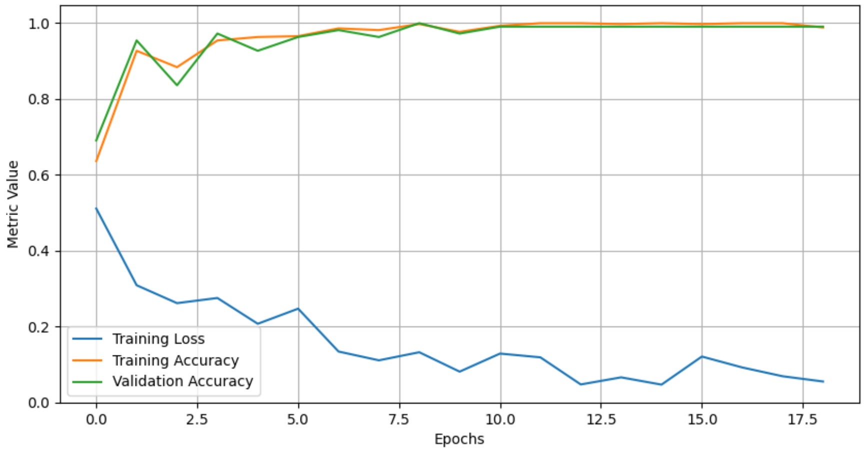
Validation

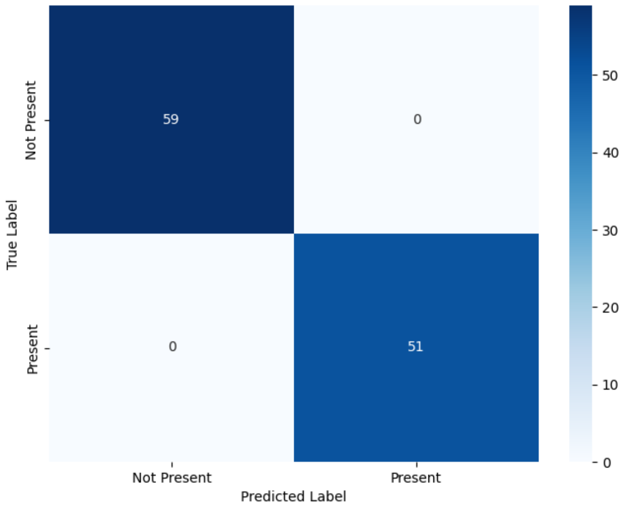

Confusion Matrix Normalized
0
background
Predicted
0.98
1.00
0.02
0
background
True
1.0
0.8
0.6
0.4
0.2
0.0


Confusion Matrix
0
background
Predicted
306
55
7
0
background
True
300
250
200
150
100
50
0

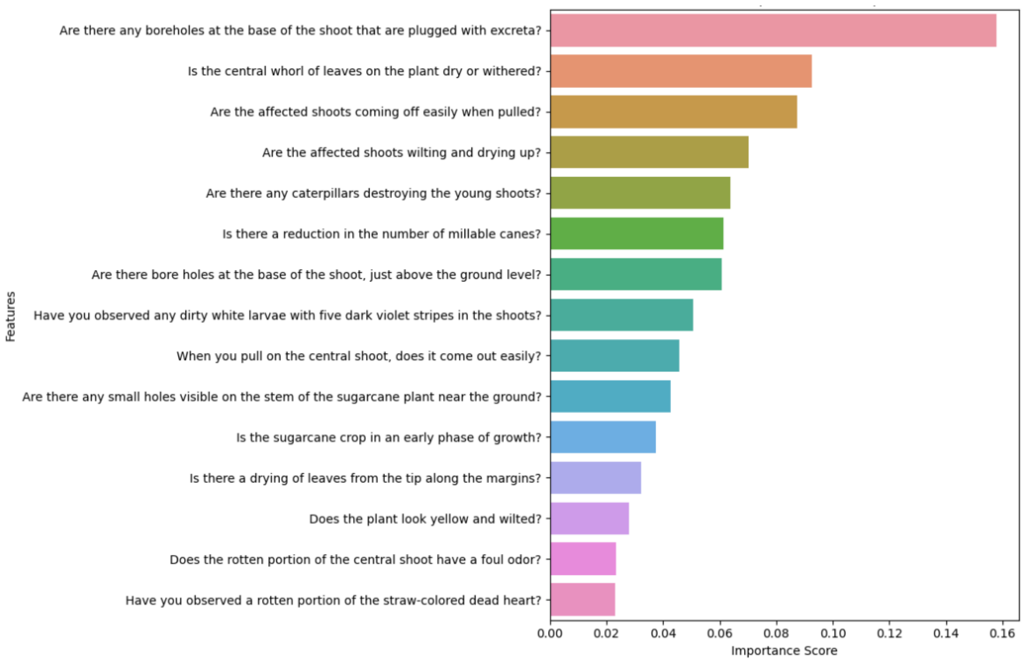
6.3.TabNet Dead Heart Model Metrics

Training History



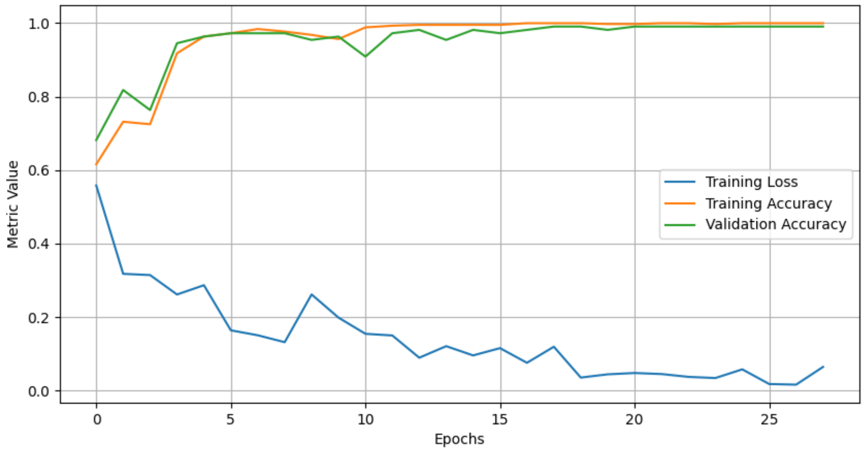


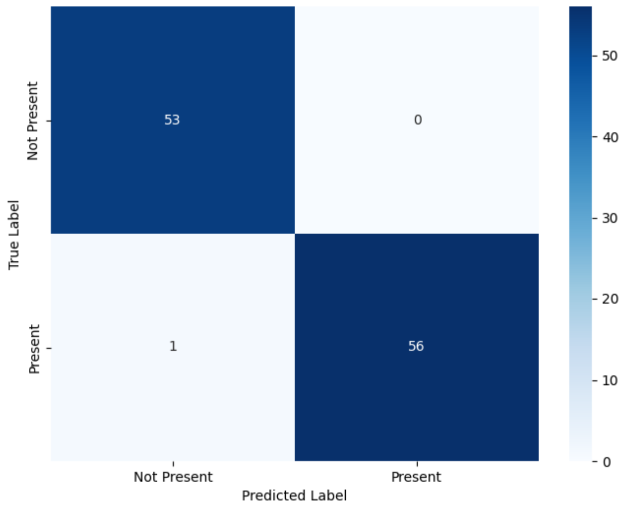
Top 15 Feature Importance



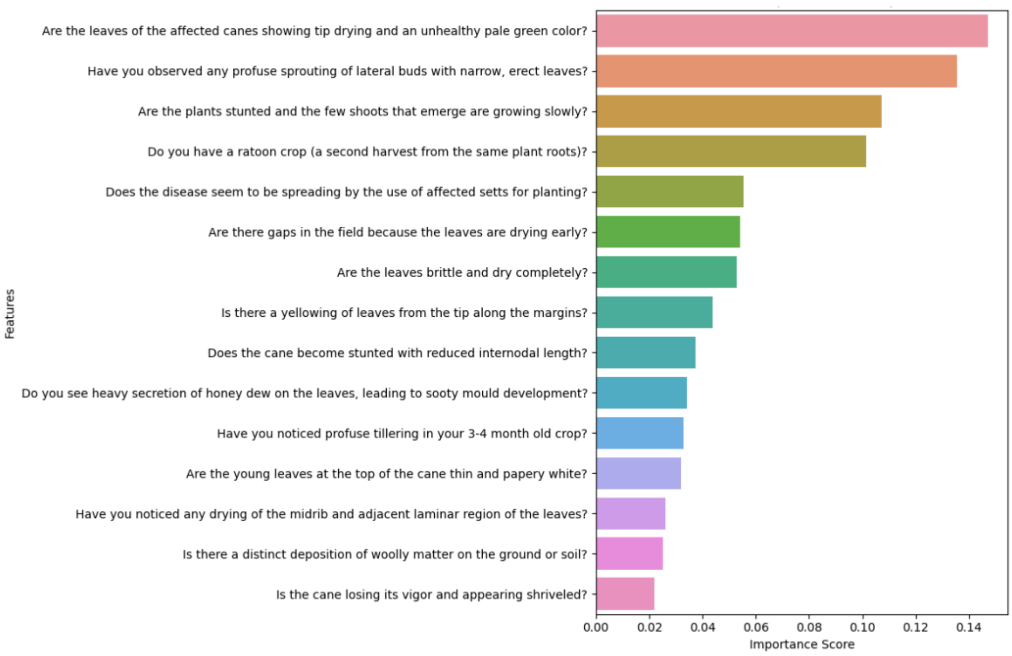
6.4.TabNet Tillet Detection Model Metrics

Training History





Top 15 Feature Importance



6.5.Model Performance Summary table

Model	Task	Dataset Size	mAP@0.5	Accuracy	Inference Time
YOLOv8s-seg	Disease Segmentation	3512 images	80.2%	—	~15.2 ms / image
YOLOv8s	Insect Detection	2925 images	97.8%	—	~13.6 ms / image
TabNet Disease Classifier	Questionnaire-Based	550 samples	—	100%	~9 ms / sample
TabNet Insect Classifier	Questionnaire-Based	550 samples	—	99.1%	~9 ms / sample

7. Application Workflow

7.1.Overview

This intelligent agricultural system combines weather data, machine learning models, and AI-powered analysis to provide farmers with accurate crop health assessments and actionable recommendations for deadheart and tiller management.

7.2.Detailed Workflow

1. Location Input & Weather Integration

Process: The workflow begins when a farmer provides their geographical location coordinates or address.

- The system uses this location data to query weather APIs for real-time and historical weather information
- Weather parameters include temperature, humidity, rainfall, wind patterns, and seasonal trends
- This environmental context is crucial for understanding crop stress factors and pest/disease pressure

2. Risk Analysis Engine

Process: The system performs comprehensive risk assessment using weather data and regional agricultural patterns.

- Analyzes current weather conditions against optimal growing parameters
- Evaluates seasonal trends and climate patterns affecting crop health
- Generates risk scores for various crop threats including deadheart disease and tiller development issues

- Creates baseline risk profiles specific to the farmer's location and current conditions

3. Detection Method Selection

Process: Farmers choose their primary concern area through an intuitive selection interface.

- **Deadheart Detection**: Focus on identifying stem borer damage and internal plant decay
- **Tiller Detection**: Emphasis on assessing tillering patterns and plant density
- This selection optimizes the subsequent analysis pipeline for the specific crop issue
- Allows for targeted model deployment and specialized recommendation generation

4. Dual Input Collection System

Image Upload Pathway

Process: Farmers upload high-quality crop images for visual analysis.

- Images should show affected plant areas, overall field conditions, or specific symptoms
- System accepts multiple image formats and automatically preprocesses them
- Images undergo quality validation to ensure they're suitable for model analysis

Questionnaire Pathway

Process: Farmers complete a structured 15-question assessment covering:

- Crop growth stage and planting date
- Observed symptoms and their severity
- Previous treatment applications
- Soil conditions and irrigation practices
- Historical pest/disease occurrences
- Current management practices

5. Advanced Machine Learning Analysis

YOLOv Model Processing (60% Weight)

Process: Computer vision analysis of uploaded images using state-of-the-art object detection.

- Identifies and localizes deadheart symptoms or tiller patterns in crop images
- Provides precise bounding boxes around affected areas
- Calculates severity scores based on visual indicators
- Generates confidence metrics for each detection
- Carries higher weight (60%) due to direct visual evidence

TabNet ML Processing (40% Weight)

Process: Structured data analysis using advanced tabular neural networks.

- Processes questionnaire responses and numerical data
- Incorporates weather data and risk analysis results
- Uses attention mechanisms to identify key predictive factors
- Generates probability scores for various outcomes
- Weighted at 40% to complement visual analysis with contextual information

6. Intelligent Prediction Fusion

Process: The system combines both model outputs using weighted averaging and ensemble techniques.

- Merges visual analysis (60%) with structured data analysis (40%)
- Applies confidence-based weighting to handle uncertain predictions
- Generates comprehensive assessment scores for crop health status
- Produces detailed breakdown of contributing factors
- Creates uncertainty estimates for transparency

7. AI-Powered Recommendation Engine

Process: Gemini AI processes all collected data to generate intelligent, contextual recommendations.

Input Integration:

- Combined prediction results from both models
- Original weather data and risk analysis
- Farmer-specific context from questionnaire responses
- Regional best practices and treatment protocols

Recommendation Generation:

- **Immediate Actions**: Urgent interventions needed within 24-48 hours
- **Short-term Management**: Actions for the next 1-2 weeks
- **Long-term Strategies**: Seasonal and crop cycle recommendations
- **Preventive Measures**: Future risk mitigation strategies
- **Treatment Options**: Specific chemical, biological, or cultural control methods
- **Monitoring Protocols**: Follow-up assessment schedules

8. Comprehensive Results Package

Process: The system compiles all analysis results and recommendations into actionable formats.

Downloadable Content Includes:

- **Executive Summary**: Key findings and priority actions
- **Detailed Analysis Report**: Complete assessment with confidence scores
- **Treatment Recommendations**: Step-by-step action plans with timing
- **Monitoring Schedule**: Follow-up assessment dates and methods
- **Weather Integration**: Optimal timing for interventions based on forecasts
- **Cost-Benefit Analysis**: Expected outcomes and resource requirements
- **Visual Documentation**: Annotated images showing detected issues

Key System Advantages

Accuracy Through Diversity

- Combines visual AI (objective) with contextual analysis (subjective)

- Weather integration provides environmental context
- Weighted ensemble approach maximizes prediction reliability

Farmer-Centric Design

- Simple interface accommodating various technology comfort levels
- Flexible input methods (images and/or questionnaire)
- Clear, actionable recommendations in accessible language

Scalability and Adaptability

- Location-aware recommendations suitable for diverse growing regions
- Continuous learning from farmer feedback and outcomes
- Integration capability with existing farm management systems

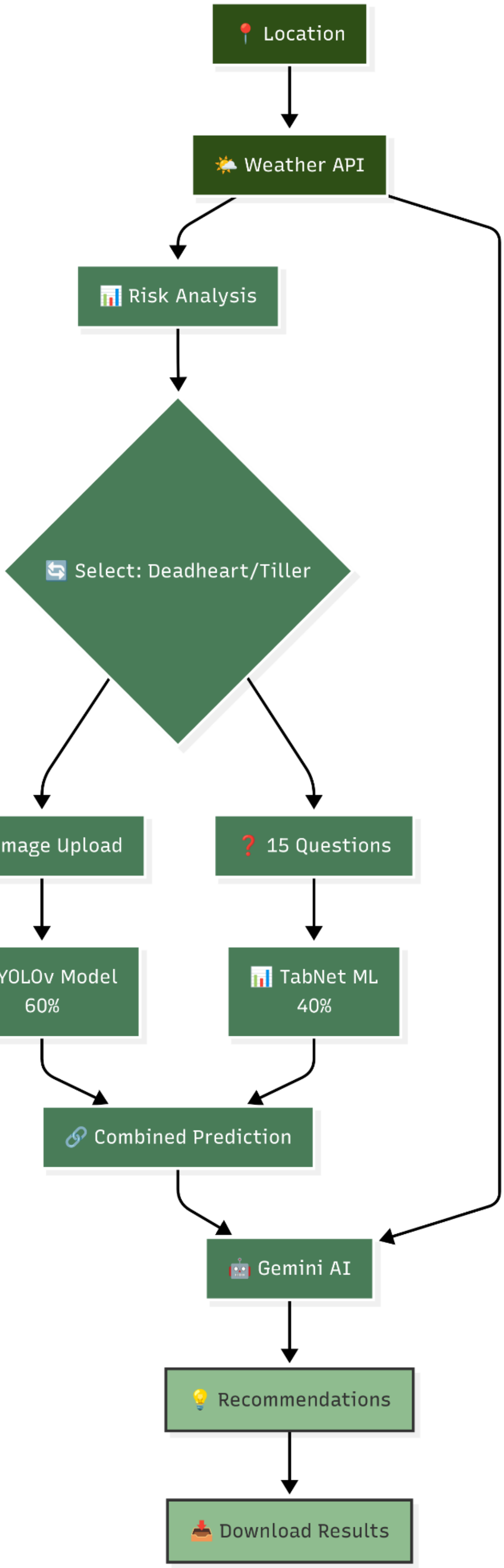
Real-Time Intelligence

- Current weather data integration for timely interventions
- Immediate analysis and recommendation generation
- Download capability for offline reference and record-keeping

Expected Outcomes

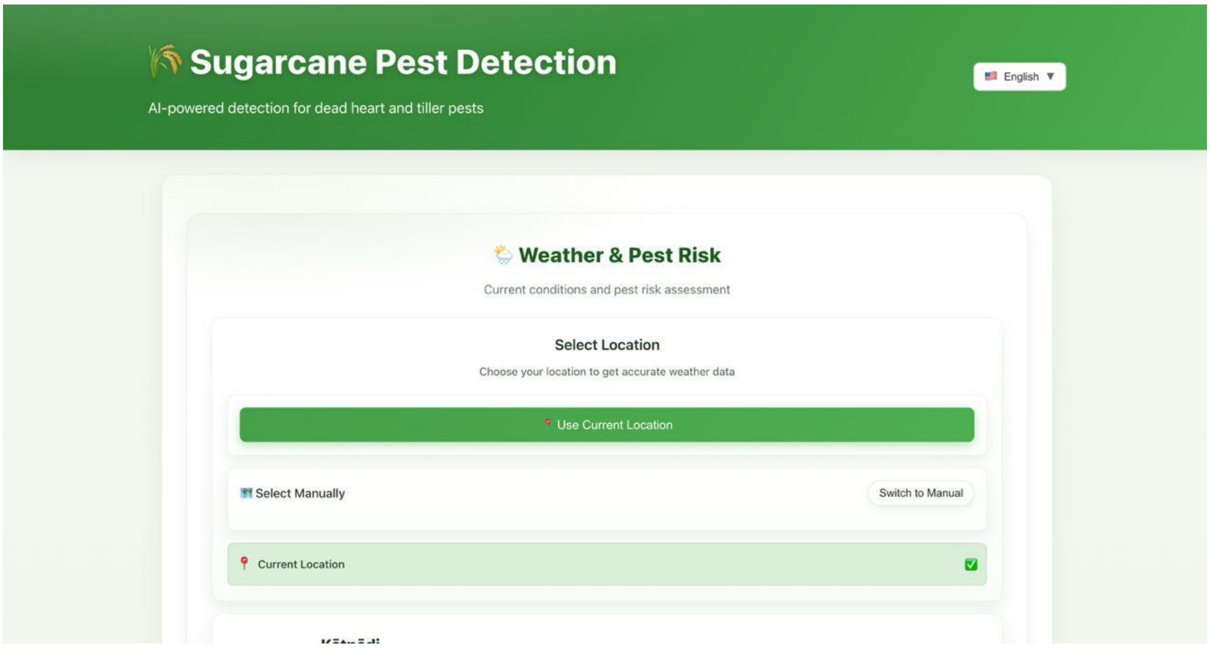
This comprehensive workflow enables farmers to make data-driven decisions about crop health management, potentially improving yield outcomes while reducing unnecessary pesticide applications and optimizing resource utilization. The system bridges the gap between advanced agricultural technology and practical farm-level implementation.

7.3.Application Workflow Diagram

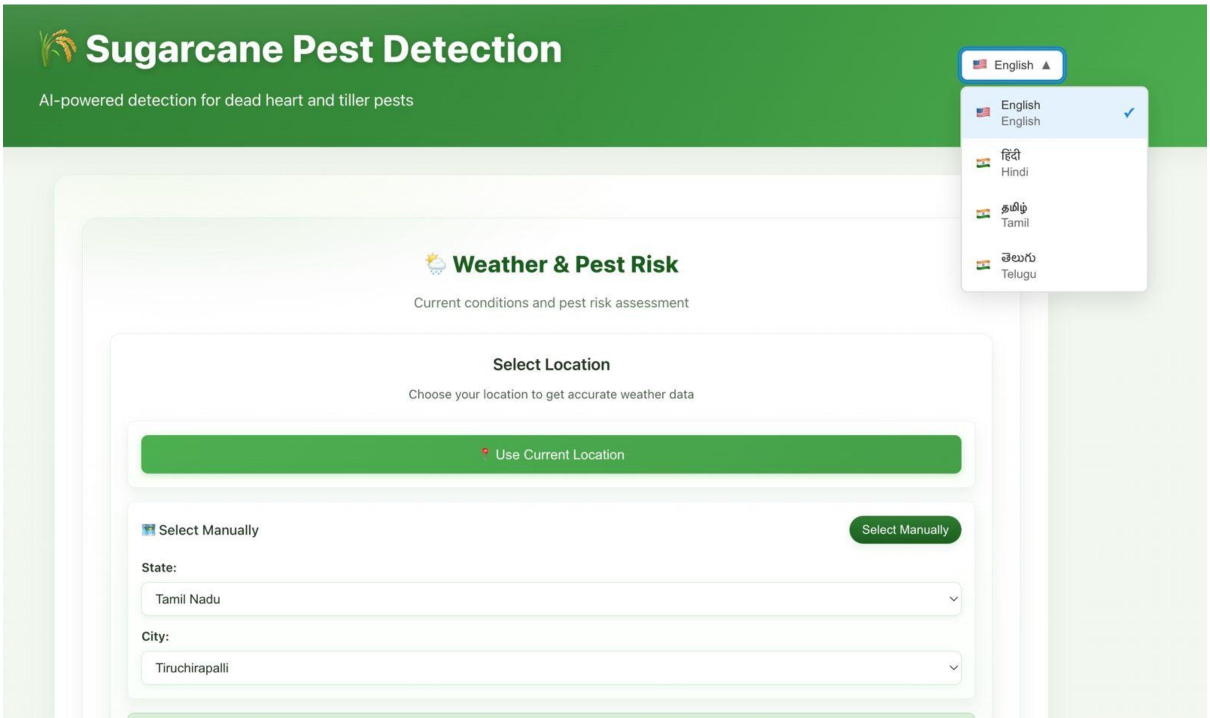


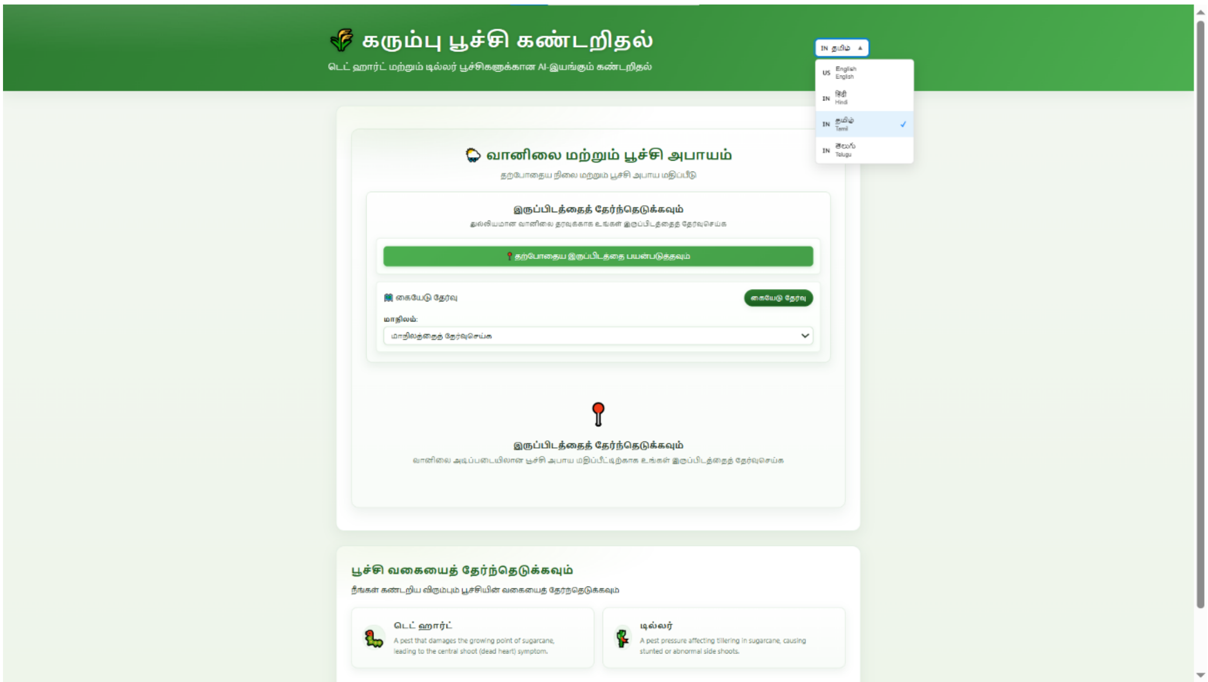
8. GUI Layout

Main Screen

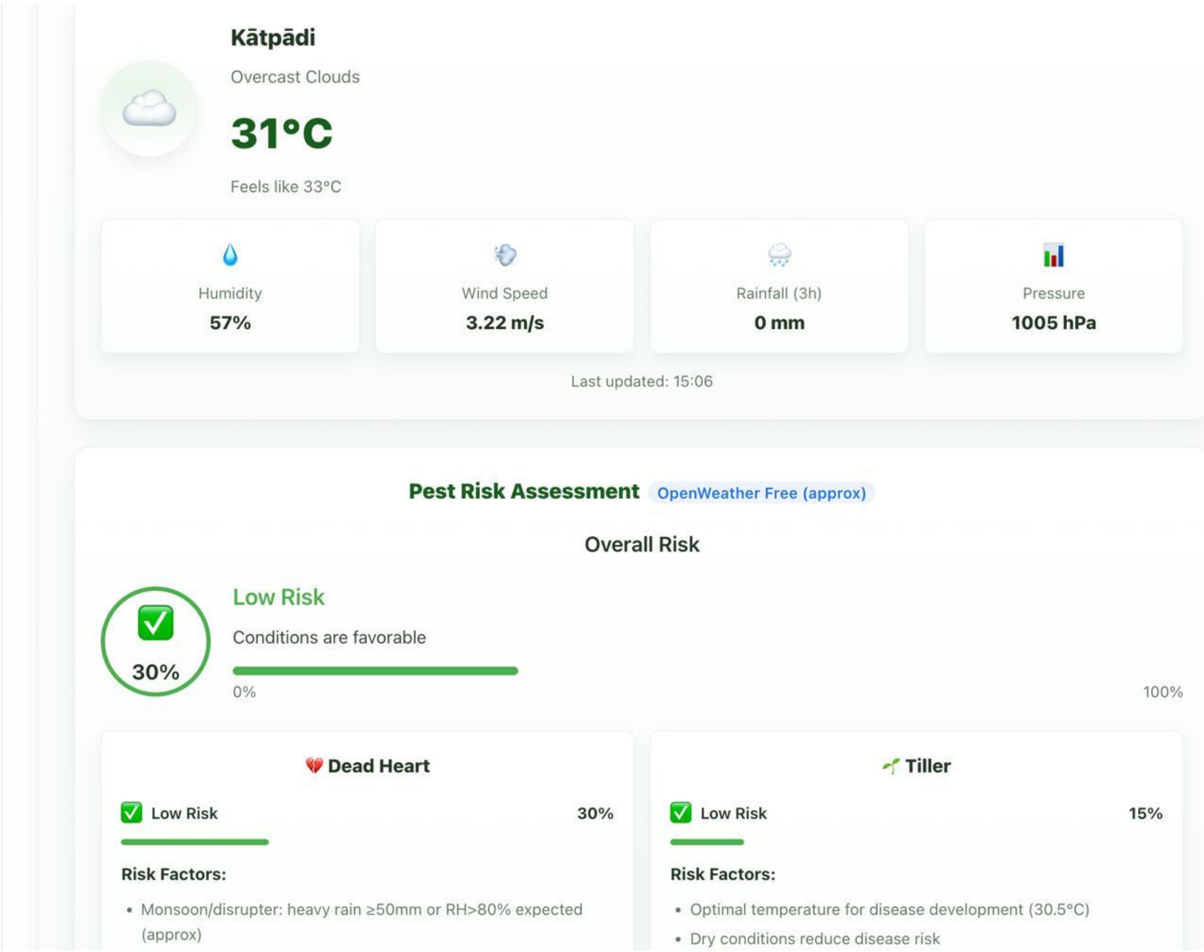


Multi-Lingual Support

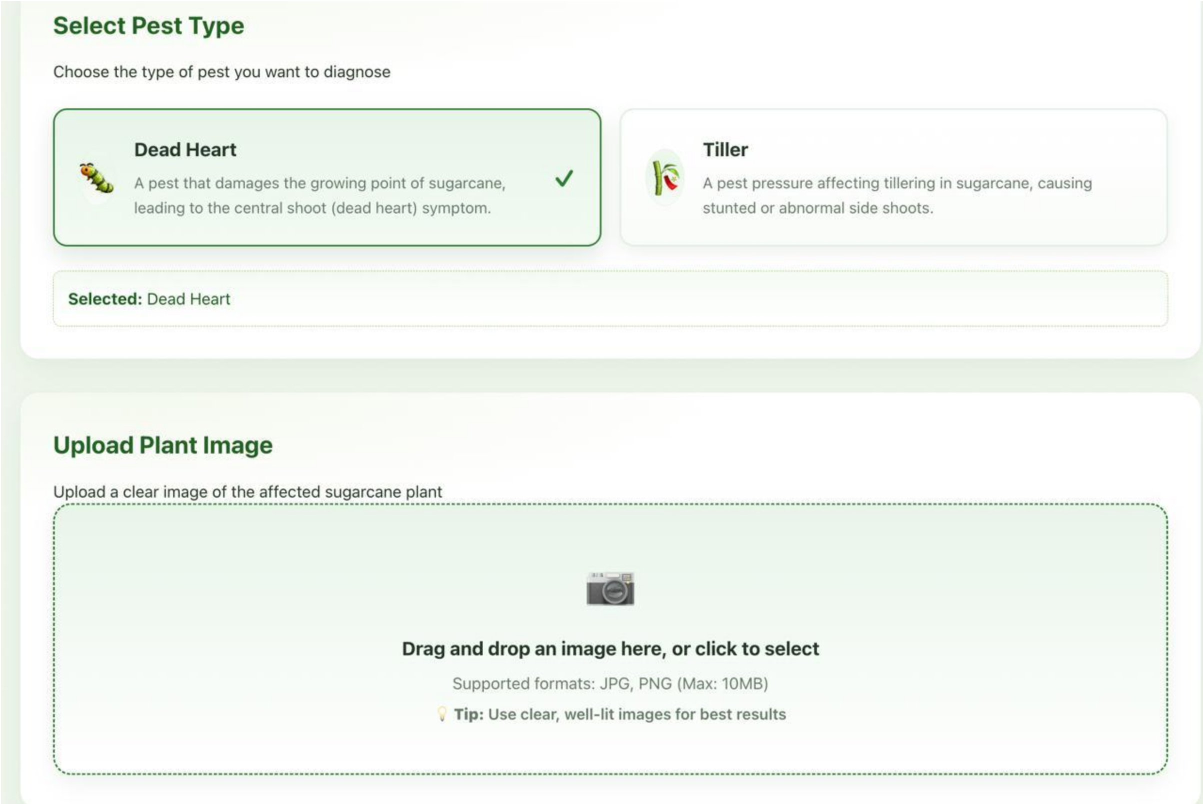




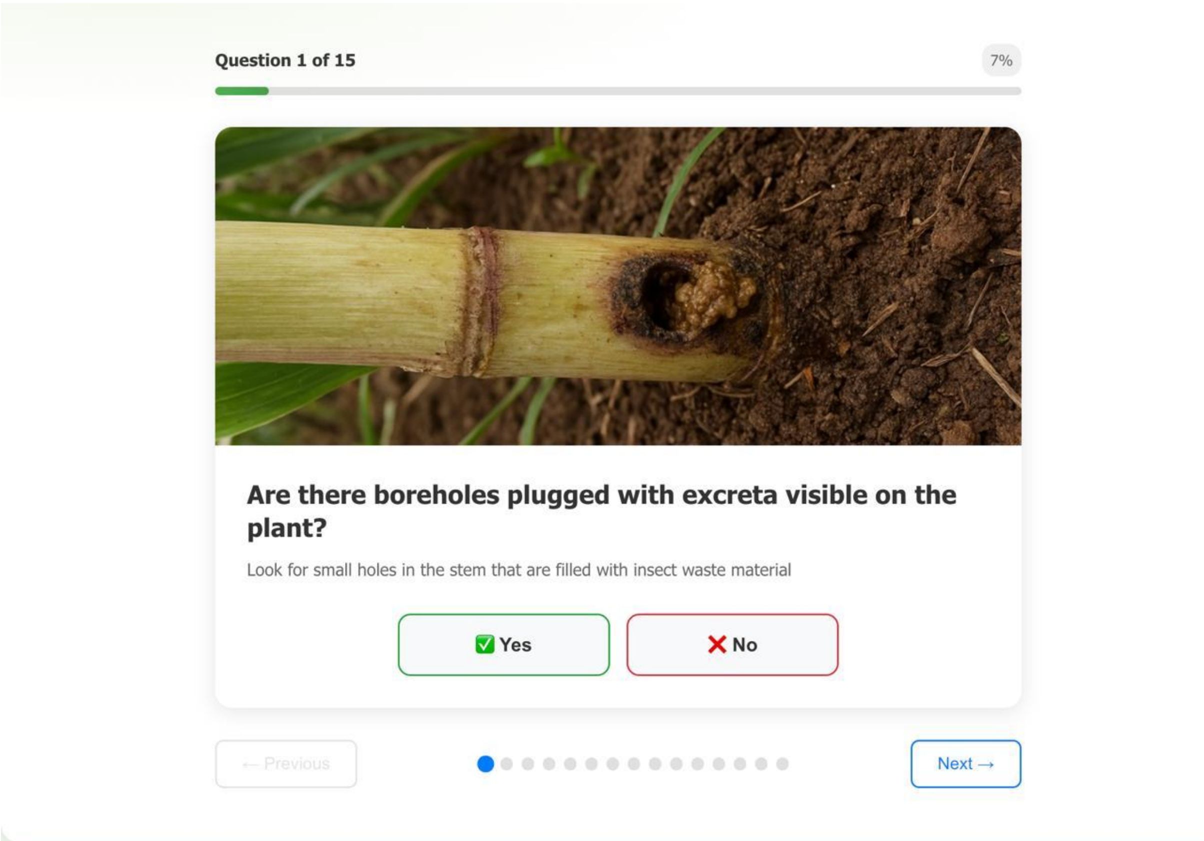
Get Current location or Specified Location weather



Upload Image



Questionnaire (Top weighted question is asked)



Analysis Result

🔬 Analysis Results

⚠️ **DEADHEART DETECTED**

Confidence: 87.9% (high)

📸 **Analyzed Image**

Score Breakdown

📊 Score Breakdown

Image Analysis		**80.0%**
Questionnaire Assessment		**99.6%**
Final Score		**87.9%**

🎯 Detections

#	Type	Class	Confidence	Region
1	segmentation	**deadheart**	80.0%	polygon (156 points)

💡 Gemini AI RecommendationsGemini

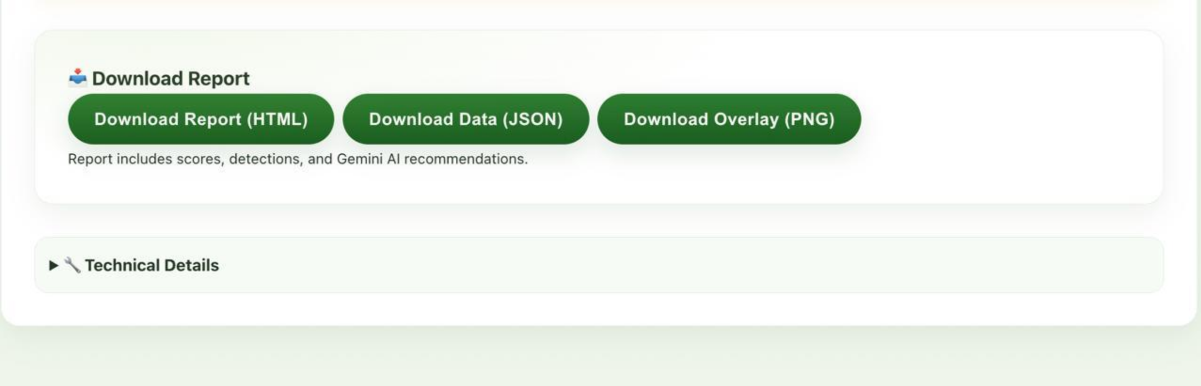
Model: gemini-1.5-flashLanguage: EN

🌿 Check your sugarcane field regularly:** Look for plants with yellowing leaves, wilting, and a rotten, straw-colored center (deadheart). Early detection is key.

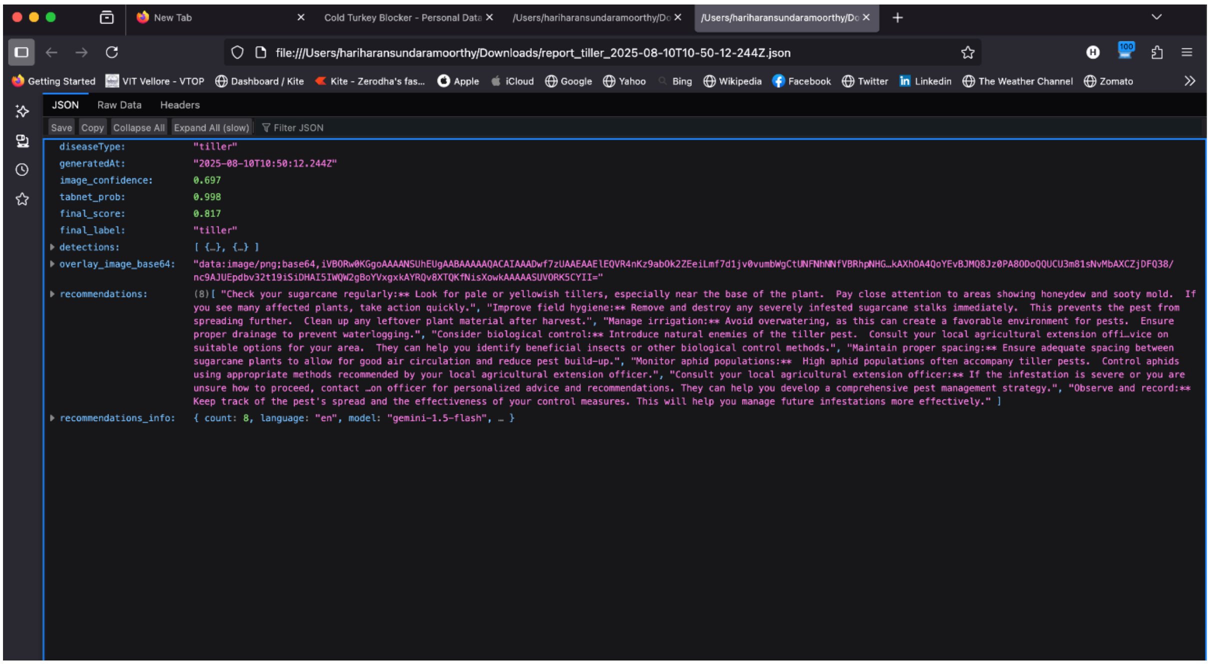
🌿 Improve field hygiene:** Remove and destroy affected plants immediately. Don't leave them in the field. This prevents the pest from spreading.

🌿 Manage irrigation:** Ensure proper drainage to avoid waterlogged conditions which favor the pest. Avoid overwatering.

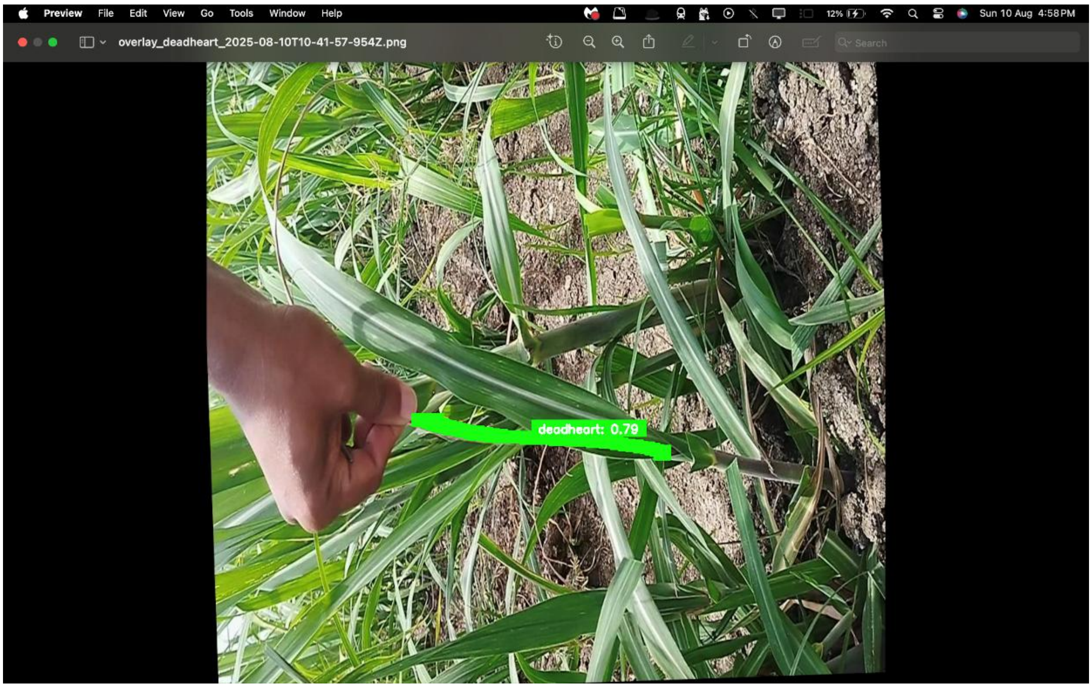
Export Format



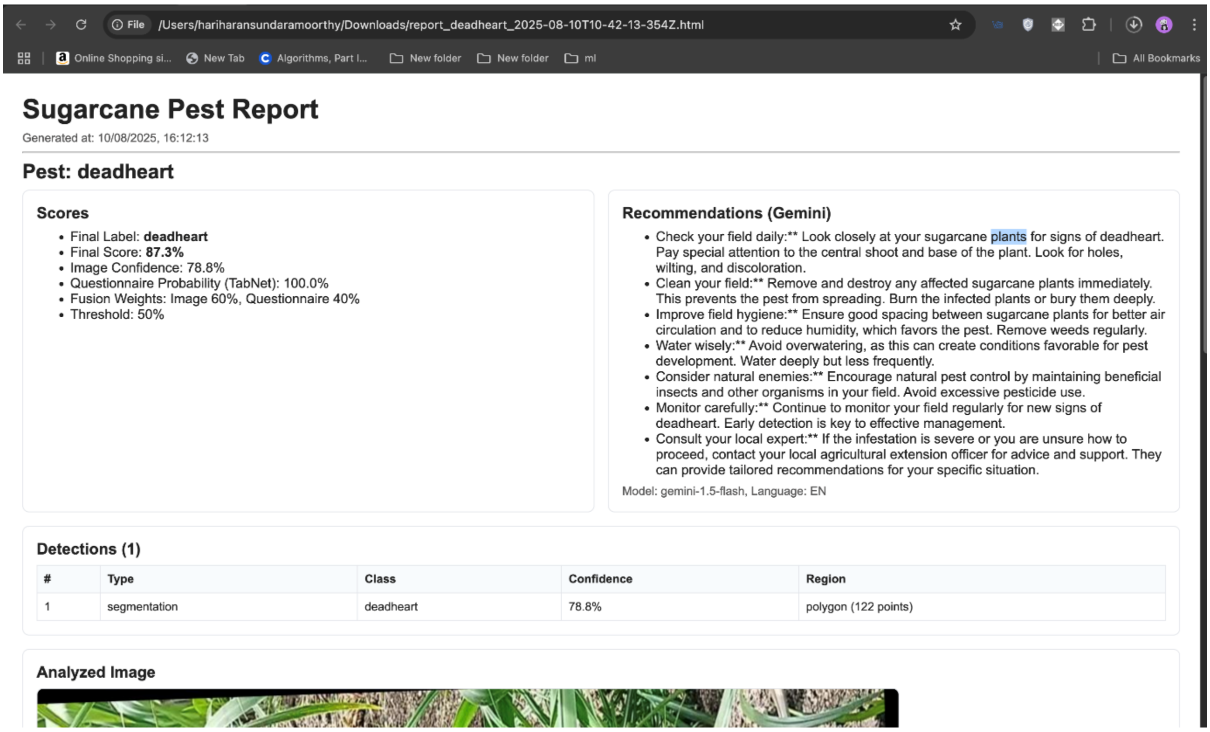
Exported JSON Format



Exported PNG



Exported HTML



Sugarcane Pest Report

Generated at: 10/08/2025, 16:12:13

Pest: deadheart

Scores

- Final Label: **deadheart**
- Final Score: **87.3%**
- Image Confidence: 78.8%
- Questionnaire Probability (TabNet): 100.0%
- Fusion Weights: Image 60%, Questionnaire 40%
- Threshold: 50%

Recommendations (Gemini)

- Check your field daily:** Look closely at your sugarcane plants for signs of deadheart. Pay special attention to the central shoot and base of the plant. Look for holes, wilting, and discoloration.
- Clean your field:** Remove and destroy any affected sugarcane plants immediately. This prevents the pest from spreading. Burn the infected plants or bury them deeply.
- Improve field hygiene:** Ensure good spacing between sugarcane plants for better air circulation and to reduce humidity, which favors the pest. Remove weeds regularly.
- Water wisely:** Avoid overwatering, as this can create conditions favorable for pest development. Water deeply but less frequently.
- Consider natural enemies:** Encourage natural pest control by maintaining beneficial insects and other organisms in your field. Avoid excessive pesticide use.
- Monitor carefully:** Continue to monitor your field regularly for new signs of deadheart. Early detection is key to effective management.
- Consult your local expert:** If the infestation is severe or you are unsure how to proceed, contact your local agricultural extension officer for advice and support. They can provide tailored recommendations for your specific situation.

Model: gemini-1.5-flash, Language: EN

Detections (1)

#	Type	Class	Confidence	Region
1	segmentation	deadheart	78.8%	polygon (122 points)

Analyzed Image

9. Running the Application

This section explains how to start the **Sugarcane Pest Detection System** in different environments. Open Application folder if not run the following GitHub repo by cloning

https://github.com/Hari20032005/sugarcane_disease_detection

9.1. Backend (FastAPI Server)

The backend runs the AI models (YOLOv8 + TabNet) and exposes REST APIs.

Steps:

1. Open a terminal and navigate to the backend/ directory:

```
cd backend
```

2. Ensure your Python virtual environment is activated:
   - **Windows**:

```
venv\Scripts\activate
```

   - **macOS/Linux**:

```
source venv/bin/activate
```

3. Start the FastAPI server:

```
uvicorn app.main:app --reload --host 0.0.0.0 --port 8000The backend API will be available at:
```

   - API Base URL: **http://localhost:8000**
   - API Documentation: **http://localhost:8000/docs**

9.2. Frontend (React Application)

The frontend provides the web-based interface for farmers and users.

Steps:

1. Open a **new terminal** (leave the backend running).
2. Navigate to the frontend/ directory:

```
cd frontend
```

3. Start the development server:

```
npm run dev
```

4. Access the application in a browser:
   - URL: **http://localhost:5173**

9.3.Full System via Docker

Docker allows running both backend and frontend together.

Steps:

1. Ensure Docker and Docker Compose are installed.
2. From the project root, run:

```
docker-compose up --build
```

3. To run in the background:

```
docker-compose up -d
```

4. Access the application:
   - Frontend: **http://localhost:5173**
   - Backend Docs: **http://localhost:8000/docs**

9.4.Summary of Commands

Component	Command
Backend	uvicorn app.main:app --reload
Frontend	npm run dev
Docker	docker-compose up --build

10. Literature Survey

Author(s) & Year	Title	Methodology Used	Advantages	Limitations
Kanwar Kumar et al. (2019)	Biology of early shoot borer, Chilo infuscatellus on sugarcane	Laboratory rearing and morphological studies; field observations	Detailed life stages & morphology; quantitative biological data	Limited to one sugarcane genotype; lab conditions may differ from field
Kumar et al. (2018)	Population dynamics of early shoot borer influenced by weather	Field surveys and correlation with meteorological data	Links pest incidence to environmental factors for better timing	Regional focus (Haryana); may not extrapolate to other agroclimatic zones
TNAU Agritech Portal (2024)	Crop protection guide for Early Shoot Borer	Integrated literature synthesis; pest identification & management	Comprehensive management recommendations; easy field application	General guidelines; lacks experimental detail
David & Nandgopal (1986)	Sugarcane Entomology in India	Review of pest species distribution and symptomology	Foundational identification of key pests and symptoms	Outdated; lacks molecular or recent control methods
Bhavani (2013)	Studies on biology of sugarcane early shoot borer	Laboratory biology on artificial diet	In-depth developmental timing; fecundity data	Artificial diet may affect natural behaviour and survival
Kalariya & Radadia (2014)	Biology of sugarcane early shoot borer in South Gujarat	Field and lab combined biological observations	Localized population info; larval instar durations	Limited geographic applicability
Plantwise Knowledge Bank (2021)	Early Shoot Borer factsheet	Pest ID and control methods synthesis	Practical pest control info with photos; user friendly	Limited to pest control; lacks detailed scientific data

11. Conclusion

This sugarcane pest detection application represents a comprehensive, AI-driven solution that addresses critical agricultural challenges through intelligent technology integration. By combining multiple data sources - real-time weather conditions, visual image analysis, and expert questionnaire assessments - the system provides farmers with accurate, actionable pest detection capabilities for deadheart and tiller infestations.

Key Strengths:

The application's hybrid approach leverages the strengths of different technologies: YOLOv computer vision for precise image analysis (60% weight) and TabNet machine learning for contextual questionnaire data (40% weight). This dual-model architecture ensures robust predictions even when one data source may be limited or unclear.

The integration of weather-based risk assessment provides proactive insights, enabling farmers to understand environmental conditions that favor pest development. The Gemini AI-powered recommendation engine transforms raw detection results into practical, tailored advice for pest management strategies.

Practical Impact:

For farmers, this translates to early pest detection, reduced crop losses, optimized pesticide usage, and data-driven decision making. The multiple download formats (PDF reports, JSON data, CSV files, annotated images) ensure compatibility with various agricultural management systems and record-keeping practices.

Future Potential:

This architecture establishes a scalable foundation that can be expanded to detect additional pest types, integrate with IoT sensors, and incorporate historical crop data for predictive analytics. The modular design allows for continuous improvement of individual components without disrupting the entire system.

The application successfully bridges the gap between advanced AI technology and practical agricultural needs, empowering farmers with professional-grade pest detection capabilities previously available only to large agricultural enterprises.