PlantVillage SSL - User Manual

Warre Snaet | Howest MCT | Research Project 2025-2026

---

1. Introduction

PlantVillage SSL is a machine learning application for **plant disease classification** using semi-supervised learning. The application is designed to run efficiently on edge devices, enabling real-time plant disease detection in agricultural environments—even without internet connectivity.

Key Features

- 🌱 **38 Plant Disease Classes** – Supports tomato, apple, corn, grape, potato, and more
- ⚡ **Ultra-Fast Inference** – <2ms per image on GPU hardware
- 🔄 **Semi-Supervised Learning** – Learns from unlabeled data via pseudo-labeling
- 🖥️ **Modern GUI** – Built with Svelte 5 & Tauri for a native desktop experience
- 📴 **Fully Offline** – No cloud or internet required

---

2. Getting Started

Launching the Application

Option A: GUI Mode (Recommended)

```
cd plantvillage_ssl/gui
bun run tauri:dev
```

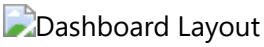
Option B: Command Line Interface

PROFI

```
cd plantvillage_ssl
./target/release/plantvillage_ssl --help
```

---

3. Dashboard Overview


Dashboard Layout

Upon startup, you are greeted by the **Dashboard**. This provides an at-a-glance view of the system:

Component	Description
Model Status	Green indicator = Model loaded. Red = No model.

Component	Description
GPU Stats	Current VRAM usage and GPU utilization
Activity Log	Recent actions (training, inference, errors)
Quick Actions	Buttons for common operations

---

4. Live Inference (Disease Detection)

This is the primary feature for diagnosing plant diseases from leaf images.

Steps

1. **Navigate** → Click the **"Inference"** tab in the sidebar
2. **Ensure Model Loaded** → Check the green status indicator
3. **Upload Image** → Click the upload area or drag-and-drop a leaf image
    - *Supported formats:* JPG, PNG
4. **View Results:**

Understanding the Results

Element	Meaning
Main Prediction	The detected disease (e.g., "Tomato Early Blight")
Confidence Score	Prediction certainty (0-100%)
Latency	Processing time in milliseconds
Top-5 Chart	Most likely disease classes visualized

Confidence Thresholds

PROFI

Color	Range	Interpretation
🟢 Green	>90%	High confidence – Safe to trust
🟡 Yellow	70-90%	Medium confidence – Verify if possible
🔴 Red	<70%	Low confidence – Manual review needed

---

5. Semi-Supervised Learning Simulation

Demonstrates how the model **learns from unlabeled data** through pseudo-labeling.

How to Run

1. **Navigate** → Click **"Simulation"** tab
2. **Configure Parameters:**
    - **Daily Batch**: Images arriving per simulated day (e.g., 100)

    - **Confidence Threshold**: Minimum confidence for pseudo-labels (default: 0.9)
    - **Retrain Threshold**: Images needed to trigger retraining (default: 200)
3. **Start** → Click **"Start Stream"**
4. **Monitor Progress:**
    - Watch pseudo-labels accumulate
    - Observe the Accepted vs. Rejected ratio chart
    - Automatic retraining triggers when threshold is reached

Key Metrics Displayed

- **Total Processed**: Images seen by the model
- **Pseudo-labels Generated**: High-confidence predictions stored
- **Rejected**: Low-confidence images (below threshold)
- **Retrain Count**: Number of retraining cycles completed

---

6. Benchmarking

Measures and validates the performance of the application on your hardware.

How to Run

1. **Navigate** → Click **"Benchmark"** tab
2. **Run** → Click **"Run Benchmark"**
3. **View Results:**
    - **Latency**: Average time per image (ms)
    - **Throughput**: Images processed per second (FPS)
    - **Comparison Chart**: Your results vs. PyTorch reference baseline

Expected Performance

Metric	Desktop GPU
Latency	~1.3 ms
Throughput	~800 FPS

PROFI

---

7. Training a New Model

From the GUI

1. Navigate to **"Training"** tab
2. Select the dataset directory
3. Configure:
    - **Epochs**: 30 (recommended)
    - **Labeled Ratio**: 0.2 (20% labeled data)
    - **CUDA**: Enable for GPU acceleration
4. Click **"Start Training"**

5. Monitor the training progress and loss curves

From CLI (Advanced)

```
cd plantvillage_ssl
cargo run --release --bin plantvillage_ssl -- train \
    --epochs 30 \
    --cuda \
    --labeled-ratio 0.2
```

---

8. Data Requirements

Dataset Structure

```
data/plantvillage/
├── train/
│   ├── Apple___Apple_scab/
│   │   ├── image001.jpg
│   │   └── ...
│   ├── Tomato___Early_blight/
│   └── ... (38 classes)
└── valid/
    ├── Apple___Apple_scab/
    └── ... (38 classes)
```

Supported Formats

- **Images**: JPG, PNG, JPEG
- **Resolution**: Any (automatically resized to 128×128)
- **Color**: RGB required

PROFI

---

9. CLI Reference

Command	Description
`train`	Train a new model with semi-supervised learning
`infer`	Run inference on a single image
`simulate`	Run SSL simulation pipeline
`benchmark`	Test inference performance
`stats`	Display dataset statistics
`export`	Export model metrics to CSV/JSON

Example Commands:

```
# View help
./target/release/plantvillage_ssl --help

# Run inference on an image
./target/release/plantvillage_ssl infer \
    --model-path best_model.mpk \
    --image-path test_leaf.jpg

# View dataset statistics
./target/release/plantvillage_ssl stats --data-dir data/plantvillage
```

---

10. Troubleshooting

Common Issues

Problem	Solution
"No Model Loaded"	Train a model first or load `best_model.mpk`
"CUDA not found"	Add CUDA to PATH: `export PATH=/usr/local/cuda/bin:$PATH`
Slow interface	Close other GPU applications, reduce batch size
"Inference Failed"	Check image format (JPG/PNG only), verify VRAM usage
Out of Memory	Reduce batch size or use CPU fallback

Checking System Status

PROFI

```
# Check CUDA availability
nvidia-smi

# Check disk space
df -h

# Monitor GPU usage
watch -n 1 nvidia-smi
```

---

11. Credits

Developer: Warre Snaet
Institution: Howest University of Applied Sciences
Program: MCT (Multimedia & Creative Technologies)
Project: Research Project 2025-2026

Technologies Used:

- Rust + Burn ML Framework
- Tauri (Desktop Application Framework)
- Svelte 5 + TailwindCSS (Frontend)
- NVIDIA CUDA (GPU Acceleration)

Document Version: 1.0 | Last Updated: January 2026