TrapTracker Desktop

User Manual

Version 0.1.1.0

1. Introduction

TrapTracker Desktop is a Windows application for **batch-processing camera-trap images** with local AI models. It lets you:

- Select a folder of images (including subfolders)
- Run object detection (YOLO-style ONNX) on each image
- Save **classified** images with bounding boxes into a mirrored `classified/` structure
- Write a **detections.csv** with results for downstream analysis
- Review results live (per-image status, class histogram, KPIs)
- Open individual images in a **Popout Viewer** (rotate, inspect)
- Explore results in an **OD Analyse** window (table, preview, charts)

This desktop tool is designed for conservationists, ecologists and projects that want **fast, offline** AI processing on stored imagery.

2. System Requirements

- **OS:** Windows 10 or Windows 11 (x64)
- **.NET:** .NET 9 Desktop Runtime (if not bundled)
- **Hardware:** CPU; optional NVIDIA GPU (CUDA) for acceleration
- **Disk space:** Depends on your image volume (originals + classified JPGs)

3. Application Overview

The **Batch Image Analysis** workspace is split into three areas:

1. **Image List (left, top):** every discovered image with per-item status ("Pending", "Processed", "No detections", error states).
2. **Summary & Histogram (left, bottom):** KPIs (totals, averages, elapsed time) and "Detections by class" bar indicators.
3. **Control Panel (right):** project/camera metadata, model picker, **CPU/GPU** provider, confidence threshold, and action buttons:
    - **Select Folder** – choose the root folder of images
    - **Classify** – run the model on all found images
    - **Analyse** – open the **OD Analyse** window on the generated CSV
    - **Cancel** – gracefully stop an in-progress run

The UI adapts for narrow layouts (same controls; stacked).

4. Running a Batch Classification

1. **Select Folder**
   Choose a root folder. The app recursively finds images (`.jpg, .jpeg, .png, .bmp, .gif`).

The app **ignores** any files under a child `classified/` directory so you can safely re-run on the same root.

2. **Enter Project / Camera ID**
   These two fields are written to the CSV for traceability.
3. **Pick Model & Provider**
   - **Model:** choose from the detected model folders.
   - **Execution Provider: CPU** or **GPU (CUDA)**. If CUDA isn't available, the app falls back to CPU.
4. **Set Confidence Threshold**
   Typical defaults: **0.50**.
5. **Classify**
   The app processes images **one by one** without freezing the UI:
   - Status column updates per image
   - KPIs (Processed, Matches, Errors, Avg Confidence, Total Detections) update live
   - The **Class Histogram** updates as detections arrive
   - A **Cancel** button is available throughout
6. **Outputs Produced**
   - **Classified images**: saved as **JPG** under a mirrored `classified/` tree that matches the original folder structure
   - **detections.csv**: written to the **selected folder** root

5. Output Structure

If your selected folder is:

```
C:\Data\CameraTraps\
  day1\img0001.jpg
  day1\img0002.jpg
  day2\img1015.jpg
```

The app writes:

```
C:\Data\CameraTraps\
  detections.csv
  classified\
    day1\img0001.jpg       (annotated)
    day1\img0002.jpg       (annotated)
    day2\img1015.jpg       (annotated)
```

- **Classified** images are **always JPG**, preserving the original filename.
- Originals remain untouched in their original locations.

6. CSV Format

A single CSV is created at the root of the selected folder:

`detections.csv` – **UTF-8**, comma-separated, with header:

```
project,camera_id,status,detections,imagename,datetime,imageurl,conf
idence_score
```

- **project:** value from UI

- **camera_id:** value from UI
- **status:** `processed`, `blank`, or `error`
- **detections:** comma-separated class labels for that image (empty if none)
- **imagename:** the original filename (e.g., `img0001.jpg`)
- **datetime:** UTC ISO-8601 at time of processing
- **imageurl:** path to the **classified JPG** if detections were saved; otherwise original path
- **confidence_score: max** confidence value among detections for that image (either `0.00–1.00` or `%` if you choose to format that way)

7. Reviewing Results

- **Image List:**
  Click the **Image Name** to open a **Popout Viewer** for the **best available** display (classified JPG when present; otherwise the original).
- **Summary & Histogram:**
  Live counts for totals, matched/no-det, error images, avg confidence (over matched), total detections, and per-class bars (scaled to total detections).
- **Analyse:**
  Opens **OD Analyse** to explore the CSV in more detail.

8. Popout Viewer

When you click an item's **Image Name** (or the preview in OD Analyse), the **Popout Window** opens:

- **Rotate Left / Right** (90° steps)
- Image scales to fit the window (maintains aspect ratio)

Optional enhancements (zoom/pan, keyboard next/previous) can be added later; this manual documents the current rotation + preview behaviour to avoid confusion.

8. OD Analyse (Detections Explorer)

Choose **Analyse** to open **OD Analyse** with the current `detections.csv`.

Features:

- **Rows table:** image name, detections, timestamp, confidence
- **Preview pane:** shows the classified JPG if present, else the original; clicking the preview opens the **Popout Viewer**
- **Filter by class:** quickly subset the rows
- **Charts:**
  - **Top Classes:** frequency of the most common detections (Top 10)
  - **Confidence Used:** binned distribution of `confidence_score` across rows

9. Tips & Best Practices

- Keep **camera names** and **project names** consistent
- Re-runs on the same folder are supported (the app ignores existing `classified/`)
- Use a **moderate confidence threshold** (0.4–0.6) initially; adjust based on results
- Back up the folder when you're happy with a run (classified JPGs + CSV are your artefacts)

---

10. Troubleshooting

No models listed

- Ensure `onnx/<ModelName>/best.onnx` exists
- Check file permissions or try running the app as a user with read access

CUDA not used

- The app falls back to CPU if CUDA isn't available; verify CUDA/cuDNN/driver installs

Images not found or zero processed

- Confirm you selected the correct **root** and that supported extensions exist
- Ensure you didn't select the `classified/` folder; the app skips it intentionally

Blank images / No detections

- Try lowering the confidence threshold
- Verify the class actually exists in your model's labels

Popout won't open

- The app attempts to open the **classified** JPG first (if created); if not found, it falls back to the original. Confirm the file path still exists and is readable.

11. Data Storage & Paths

- **Classified files** are written adjacent to your selected root folder in a mirrored `classified/` tree
- **CSV** is written next to the selected root as `detections.csv`
- The app does **not** move or delete your originals

12. Updates

When using a newer app build:

- Your outputs (CSV + classified JPGs) live in your chosen data folders and remain intact

13. Support

- **Website:** traptracker.co.uk
- **Email:** paul.fergus@gmail.com

14. Version History (Desktop)

- **0.1.1.0** – Initial public release of the batch desktop classifier and analysis tools