WildAIID
Wildlife Artificial Intelligence Identification
Verion 1.0.0

Hamza Farooq and Terry Ord
UNSW Sydney

Document version: 26 June 2025

CITIATION: **WildAIID** was created by Tariq Khan, Hamza Farooq, Erik Meijering and Terry J. Ord and should be cited as:

Khan T, Farooq H, Meijering E and Ord TJ (2025). WildAIID: Wildlife Artificial Intelligence Identification. Version 1.0.0. www.ordlab.unsw.edu.au/WildAIID

Face recognition by **WildAIID** relies on **GhostFaceNets** © 2019 Sefik Ilkin Serengil under the following conditions: Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to this copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License details -- https://github.com/serengil/deepface/blob/master/LICENSE

WildAIID also uses these other open access software:

Yolo10 used without significant modification:

Copyright © 2007 Free Software Foundation, Inc. <https://fsf.org/>
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

License details -- yolov10/LICENSE at main · THU-MIG/yolov10 · GitHub

TensorFlow:

Apache License Version 2.0, January 2004 (http://www.apache.org/licenses/)

License details -- https://github.com/tensorflow/tensorflow/blob/master/LICENSE

PyTorch:

Copyright © 2016- Facebook, Inc (Adam Paszke)
Copyright ©2014- Facebook, Inc (Soumith Chintala)
Copyright © 2011-2014 Idiap Research Institute (Ronan Collobert)
Copyright ©2012-2014 Deepmind Technologies Koray Kavukcuoglu)
Copyright ©2011-2012 NEC Laboratories America (Koray Kavukcuoglu)
Copyright © 2011-2013 NYU (Clement Farabet)
Copyright © 2006-2010 NEC Laboratories America (Ronan Collobert, Leon Bottou, Iain Melvin, Jason Weston)
Copyright ©2006 Idiap Research Institute (Samy Bengio)
Copyright © 2001-2004 Idiap Research Institute (Ronan Collobert, Samy Bengio, Johnny Mariethoz)

License details -- https://github.com/pytorch/pytorch/blob/main/LICENSE

Tkinter:

MIT License: Copyright ©2018 Packt

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

License details -- https://github.com/PacktPublishing/Python-GUI-Programming-with-Tkinter/blob/master/LICENSE

Python:

Python software and documentation are licensed under the Python Software Foundation License Version 2.

License details -- https://docs.python.org/3/license.html

Contents

1 Introduction

1.1 Overview of WildAIID

WildAIID is an advanced AI-driven platform designed to streamline the training of machine learning models using images and videos for wildlife identification. The platform supports the development, testing, and optimization of AI models aimed at enhancing wildlife conservation efforts through automation. By utilizing a variety of visual data, WildAIID aids in the identification, classification, and tracking of wildlife species, with a particular focus on kangaroos for this project.

1.2 Objective

The system assists in automating the process of training AI models by processing visual data, with capabilities to handle images, videos, and test datasets for model evaluation. Specifically, WildAIID is equipped to generate datasets tailored for kangaroo identification by extracting key visual features and creating annotated datasets for AI model training and testing. This includes automatically labeling images and videos with kangaroo-specific annotations to facilitate the development of accurate and efficient wildlife identification systems.

2 System Requirements

To ensure optimal performance and efficient operation of WildAIID, the following system specifications are recommended:

2.1 Operating System

• Windows 10 or Windows 11 (64-bit)

2.2 Processor

• **Minimum:** Intel Core i7 or AMD Ryzen 7 (or equivalent)
• **Recommended:** Intel Core i9 or AMD Ryzen 9 (or higher for optimal performance)

2.3 Memory (RAM)

• **Minimum:** 24 GB RAM
• **Recommended:** 32 GB RAM or more for smoother operation, especially when handling large datasets and video processing.

2.4 Storage

• **Minimum:** 500 GB SSD (Solid-State Drive)
• **Recommended:** 1 TB SSD or higher for faster data access and smoother operations, especially when working with large image/video datasets.

2.5 Graphics Processing Unit (GPU)

• **Minimum:** 16 GB GPU RAM (Dedicated GPU)
• **Recommended:** 24 GB or more GPU RAM for faster training and processing, especially for AI model training on large datasets.

Note: Non-GPU configurations are supported, but performance will be significantly slower—up to 22 times slower for certain tasks—due to reliance on CPU processing.

3 Download WildAIID

WildAIID is available for download from: https://www.ordlab.unsw.edu.au/WildAIID

4 Installation and Setup

WildAIID is distributed as a ready-to-run executable file, so no installation process is required. All necessary packages and dependencies have already been built and included. Simply ensure that you have the required folder structure (with the "Build" and "Dist" folders), and then follow the steps below to run the system.

Figure 1: Required Folder Structure for WildAIID.

4.1 File Structure

You need to copy or download the following two folders show in figure 1:

• **Build Folder**: This folder contains a subfolder tkinter-gui show in 2 which includes all the necessary packages and backend files required for the application.

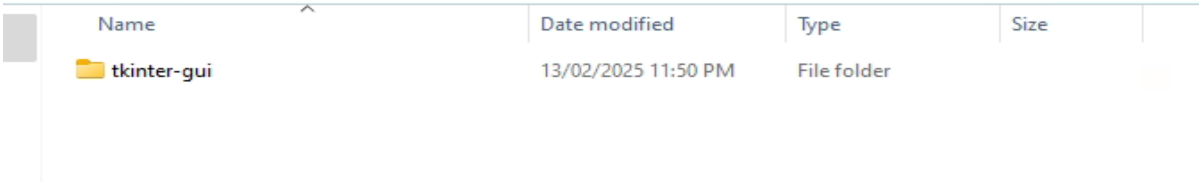

Figure 2: Build Folder must contain subfolder tkinter-gui.

• **Dist Folder**: This folder contains the other_assets folder and the executable file tkinter-gui.exe show in figure 3.

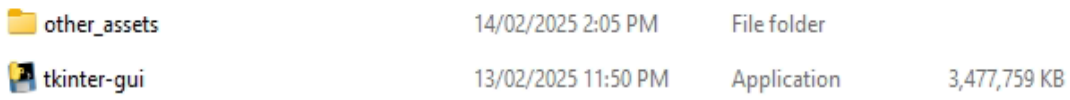

Figure 3: dist folder must contain sub folder other-assets and thinkter-gui exi.

Make sure that both folders are present and the file structure **is intact**. The most common isse that can occur is your computer's or brower's security settings block the download of large files (usually files greater than 3 gigs). Users are often not aware certain items haven't been downloaded or unzipped. The best way to determine that you have everything for WildAIID to run is by looking directly into each of the two downloaded folders and make sure the folder structure looks like the below (in particular noting that 'tkinter-gui.pkg' is present):

- Build/
  - tkinter-gui/
    * All necessary packages and backend files show in figure 4.

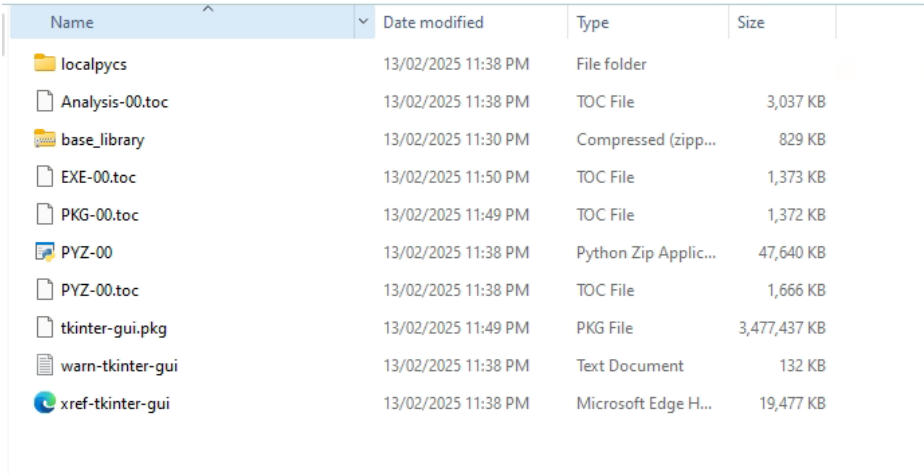

Figure 4: thinkter-gui must contain these sub folders.

- Dist/
  - other_assets/ show in figure 5

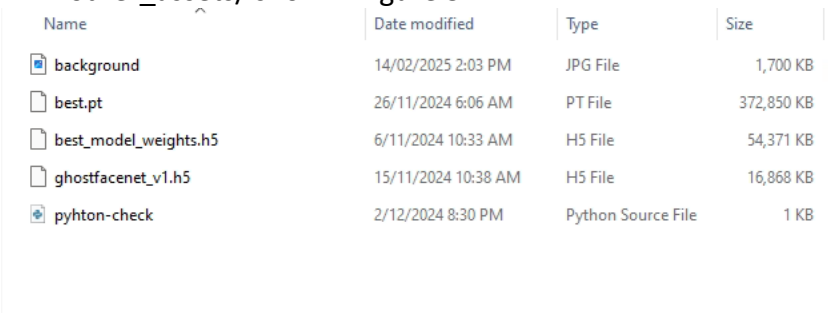

Figure 5: other assets must contain these sub folders.

- tkinter-gui.exe

Once you have both folders and confirmed the contents are present in full, you are ready to run the system!

4.2 Setting Up for GPU-based Systems

If you're running the application on a GPU-based system, you'll need to install CUDA drivers for optimal performance. Follow the steps below to install and set up CUDA on your computer. PLEASE NOTE: the links below were correct and active at time of release of WildAIID, but may need updating (search the keywords for a given download on the developer's official website).

4.2.1 CUDA Driver Installation

• First, ensure that you have a compatible NVIDIA GPU installed.

• Download and install the appropriate **CUDA Toolkit (version 11.2.0)** from the official NVIDIA website: https://developer.nvidia.com/cuda-toolkit.

• Download and install **cuDNN (version 8.1.0, for CUDA 11.0, 11.1 and 11.2)** from the official NVIDIA website: https://developer.nvidia.com/cudnn.
A note on version updates to these two applications: WildAIID uses TensorFlow, which is quite sensitive to the CUDA and cuDNN versions, and currently is the most compatible and stable with CUDA Toolkit version 11.2.0 and cuDNN version 8.1.0. It is what it is! We are actively updating WildAIID and removing this dependence with third party applications or specific versions of those third party applications is a priority for WildAIID's development. In the meantime you will be able to find and download the older versions of these applications from the NVIDIA links above. Unless you have the stated versions above, WildAIID won't recognise the applications and your GPU performance won't be incorporated into WildAIIDs utilities.

• After downloading the cuDNN version 8.1, extract it. The extracted folder should contain three folders **"bin, include, and lib"**. You must copy all three folders so that we can paste them in the CUDA v11.2 folder in the following steps. Once you have all three folders copied, open up your file explorer go to **"this PC"** select **"Local Disc"** next select **"Program Files"** next select **"NVIDIA GPU Computing Toolkit"** then select **"CUDA"** and finally select your CUDA version folder (which should be 11.2). Now paste the three folders from the cuDNN. Now we have completed all necessary items for the installation of CUDA to make full use of your GPU's capacity.

• During installation, ensure the following components are selected:
- CUDA Toolkit (necessary for GPU acceleration).
- NVIDIA Driver (compatible with your GPU model).
- cuDNN (NVIDIA's deep learning library).

• After installation, update your OS PATH environment variable to include the following directories (assuming the default installation paths):

- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA˘11.2\bin
- C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA˘11.2\libnvvp

NOTE: for Windows 11 this can be done by pressing Windows + I to open Settings, then go to System > About > Advanced system settings. Click Environment Variables. In the 'User variables for XXX' window at the top, double click on the 'Path' variable. In the Edit environment variable that pops up, select 'New…' and then copy the above two path's into the new variable boxes that appear.

• Verify that CUDA is correctly installed by running the following command in the command prompt (or terminal):

nvcc --version

This command will output the installed version of CUDA, confirming the installation.

NOTE: for Windows 11 you can access the Command Prompt terminal by clicking on the Start menu in the bottom left corner of your screen and typing "cmd" into the search bar. Select the option to open Command Prompt and then type "nvcc –version" at the command line.

• Restart your system after installation to ensure that all drivers and environment variables are properly configured.

Once CUDA and cuDNN are installed and configured, the application will leverage GPU acceleration for faster processing. If these drivers are not installed, the system will default to CPU processing, which may be significantly slower for certain tasks.

5 Launching the Application

To launch the WildAIID application, follow the steps below:

• Locate and double-click on the shortcut 'WildAIID app' (the parent executable for this shortcut is 'tkinter-gui' with the .exe extension found in the dist folder). NB: it may take a few moments for the application to open.

• Upon launching, you will first see a blank, black window. This is the Python command prompt that runs in the background, initializing the system. **NOTE**: if you are using a GPU set up on your computer, insure there are no warning messages appearing in the Python command window once the WildAIID window has eventually opened (e.g., "Could not load dynamic library 'cudart64_110.dll'; dlerror: cudart64_110.dll not found"). If you encounter an error relating to the GPU, it is either because your computer does not have a GPU system or you haven't downloaded the correct versions of CUDA and cuDNN outlined in Section 4.

• After a brief moment, the graphical user interface (GUI) shown in figure 6 of the application will appear, allowing you to interact with the system.

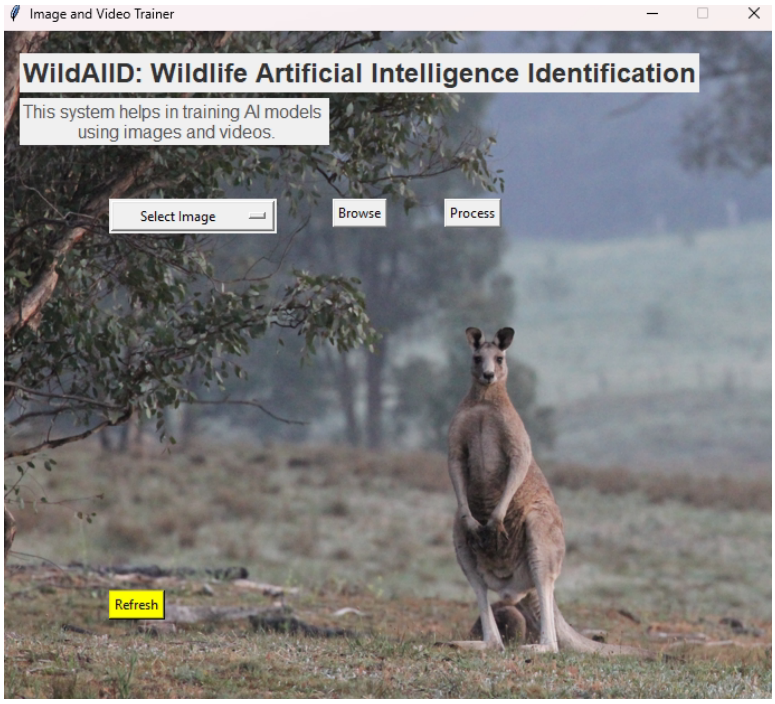

Figure 6: GUI interface for Kangaroo face recognition.

Note: The black window show in figure 7 is an essential part of the initialization process and will automatically open once the GUI interface is fully loaded and help to prompt back-end process. If the GUI interface does not open after some time, ensure that your system meets the necessary requirements and that all dependencies are correctly installed.

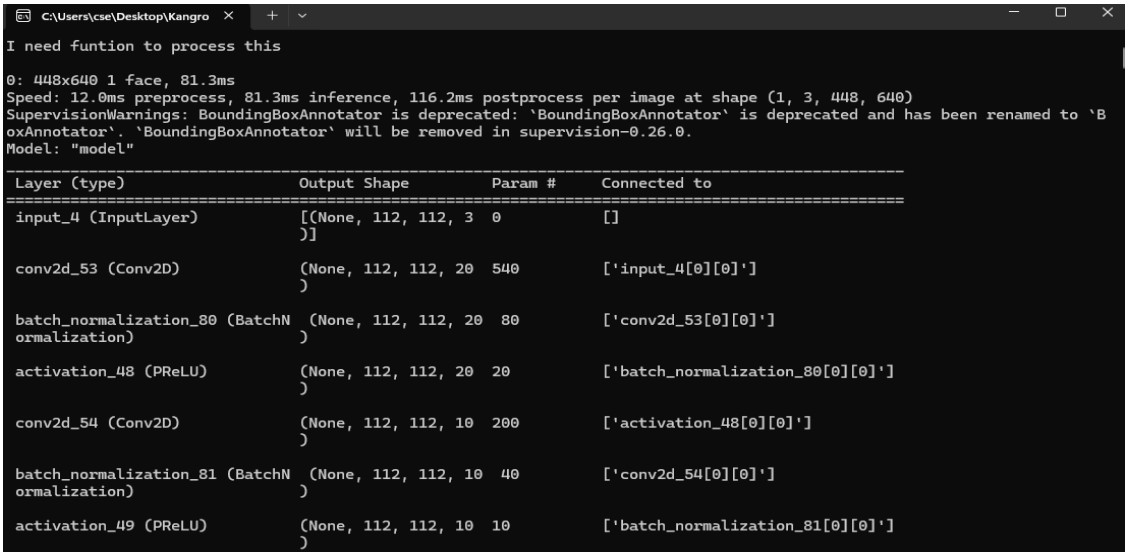

Figure 7: The black window displays back-end processes.

6 Navigating the User Interface

6.1 User Interface Overview

The WildAIID interface is designed for intuitive interaction and seamless navigation. It includes several key components:

• **Project Title and Description:** Provides a brief overview of the system's objectives and functionality.

• **Progress Bar:** Displays real-time task progress, providing feedback on the completion status.

• **Time and Output Labels:** Show the elapsed processing time and task-related output or status updates.

6.2 Interactive Elements

The user interface includes several interactive elements to facilitate task execution the figure 9 shown below:

• **Task Selection:** Select the task you want to perform from the available options.

• **Browse Folder:** Browse and select the folder containing the necessary data for the task.

• **Process Button:** After selecting the task and folder, press the *Process* button to begin the task execution.

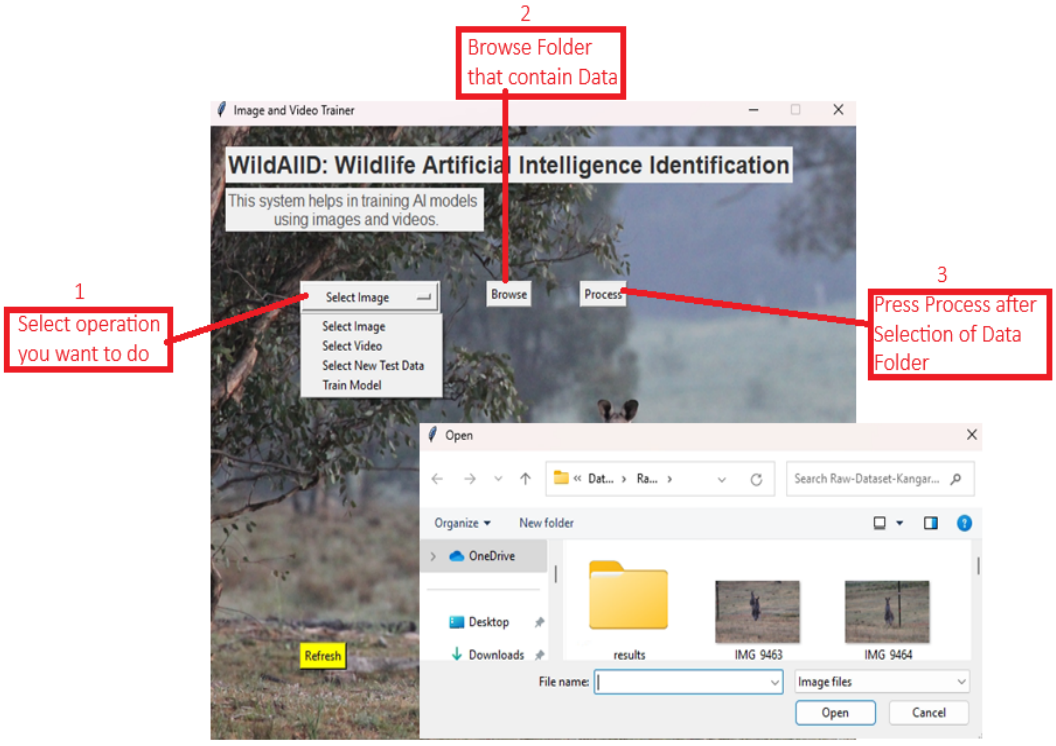

Figure 8: Interactive Elements of GUI to Start Operation.

7 Tasks And Operations

Users must first select the specific task they want to perform. Available tasks include:

- Select Image
- Select Video
- Select New Test Data
- Train Model

7.1 Select Image

The **Select Image** shown in figure option allows users to perform inference on an image, such as detecting a kangaroo in a wildlife photograph. The process involves the following steps:

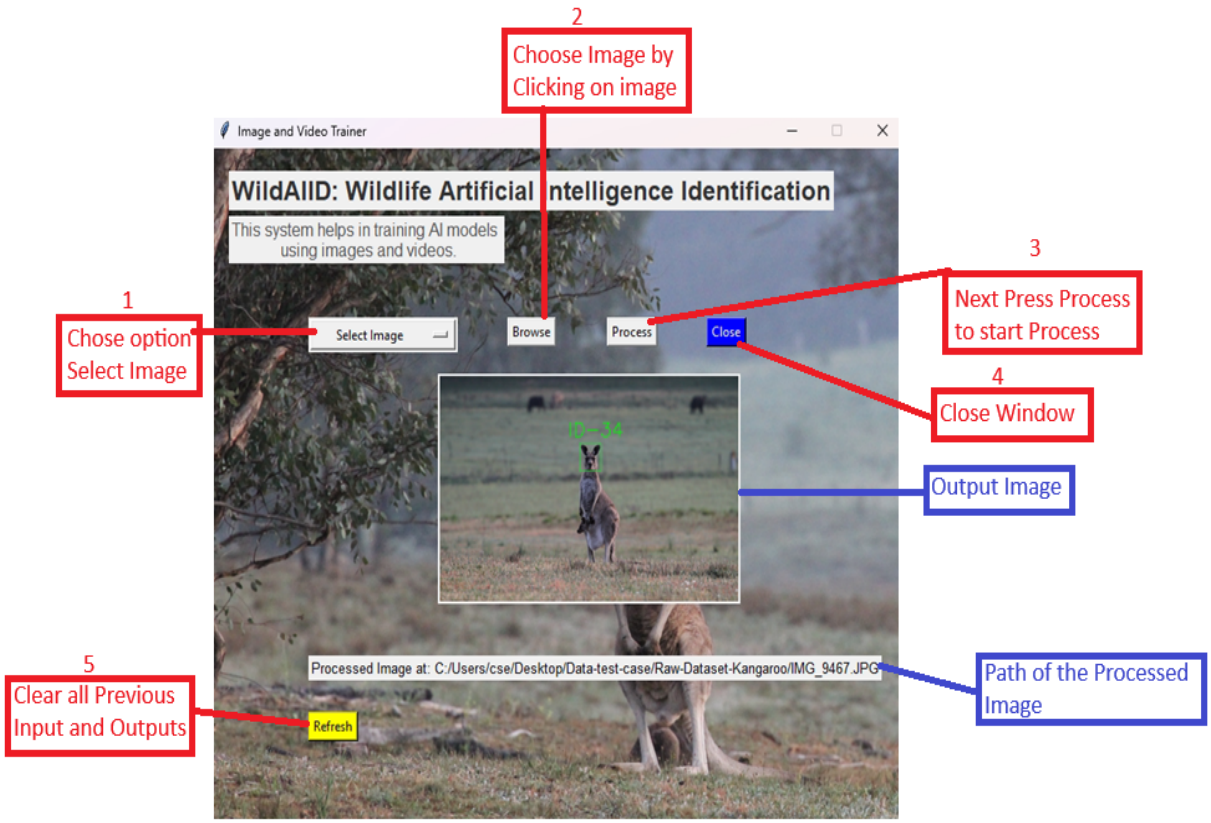

Figure 9: Output of Select Image option.

- Click on the **Select Image** option to initiate the process.
- Browse and select the desired image for inference.

• Once selected, the system processes the image and displays the output with detected objects.

• The **Refresh** button clears the displayed image, allowing users to select a new image.

• The **Close** button closes the image display window.

7.2 Select Video

The **Select Video** option, as shown in Figure 10, enables users to perform inference on a video, such as identifying wildlife activity frame by frame. The process follows these steps:

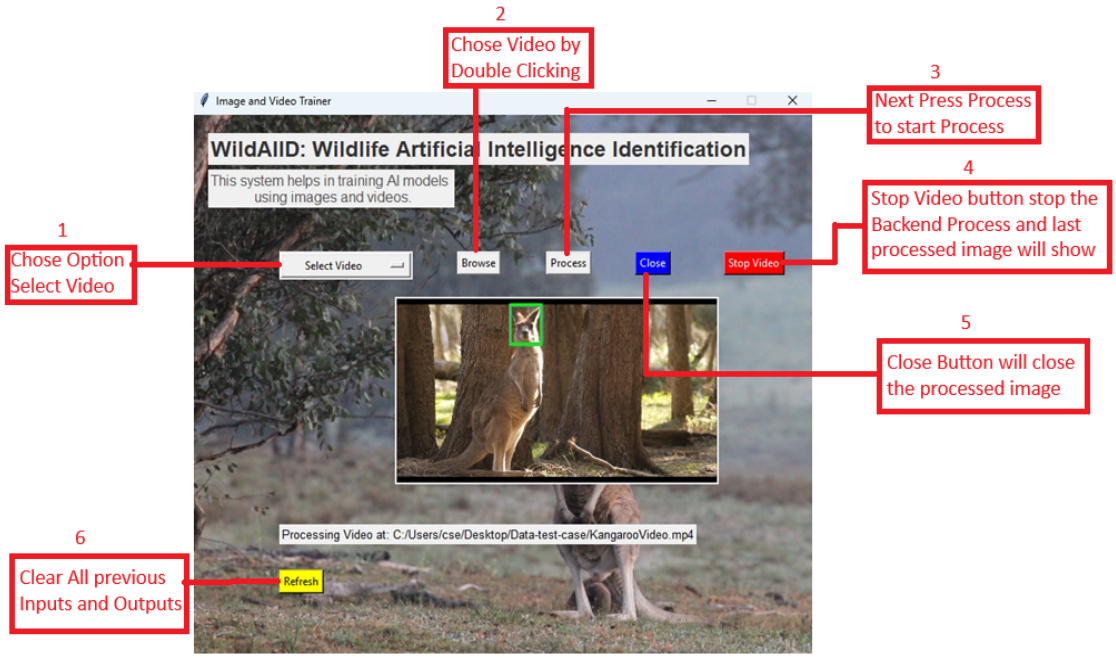

Figure 10: Output of Select Video option.

• Click on the **Select Video** option to begin.

• Browse and select the desired video file for processing.

• The system processes the video, detecting objects and displaying the results in real time.

• The **Stop** button allows users to pause processing at any time and terminates back-end process.

• The **close** button close the inference window.

• The **Refresh** button will clear all the inputs and outputs.

7.3 Test New Data

The **Test New Data** option allows users to process new datasets for inference. It provides various interactive controls to manage input selection, processing, and result visualization. The steps are as follows:

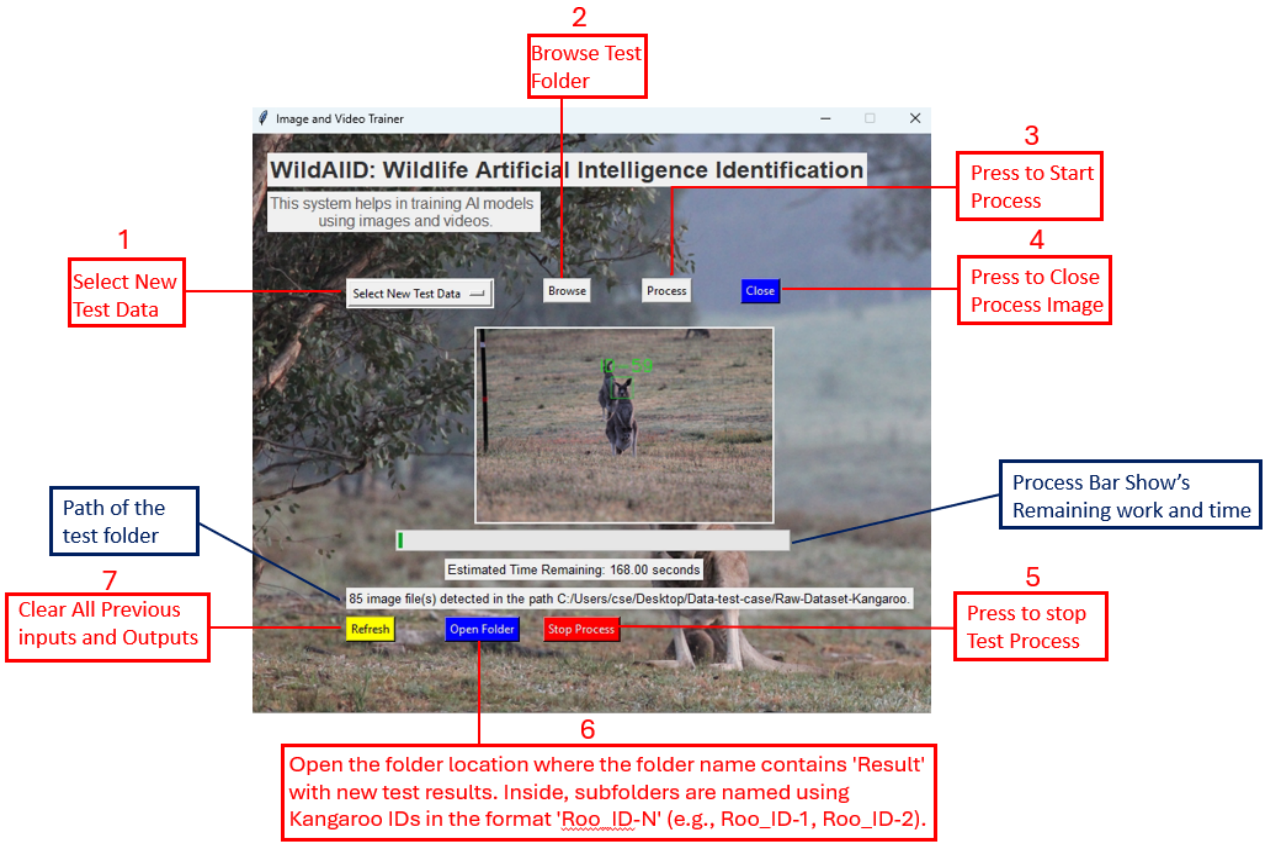

Figure 11: Output of Test New Data option.

• Click on the **Select** button to choose the type of data (image or video).

• Use the **Browse** button to navigate and select the desired dataset for inference and dataset structure in section 7.2.

• Click on the **Process** button to start analyzing the selected data.

• The **Close** button will exit the inference window.

• The **Stop** button terminates the processing at any time.

• The **Open Folder** button allows users to access the directory containing processed results.

• The **Refresh** button clears all inputs and outputs, preparing the system for a new task.

7.4 Training Data

The **Training Data** option enables users to train a model using selected datasets. It provides various interactive controls for data selection, training execution, and result management. The steps are as follows:

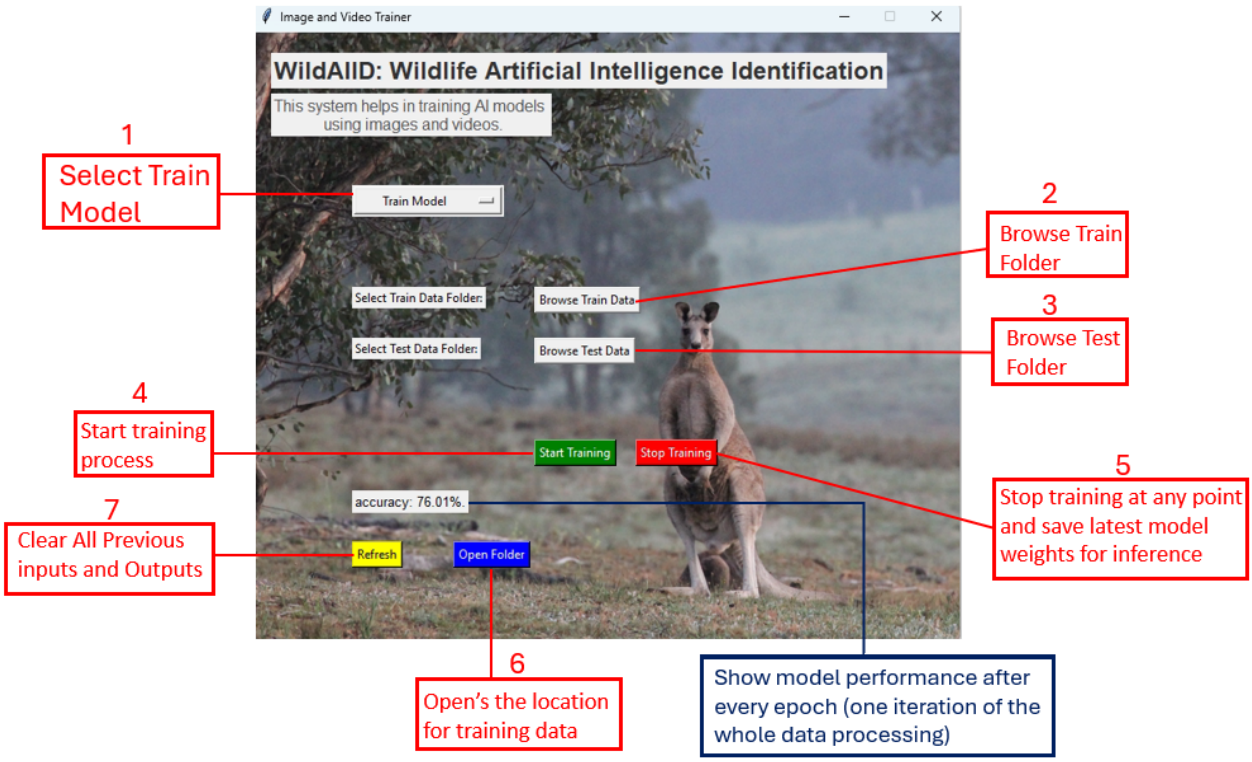

Figure 12: Training Data Interface.

• Click on the **Train Model** button to initiate the training process.

• Use the **Browse Train Data** button to select the training dataset structure is shown in section 7.1.

• Use the **Browse Test Data** button to select the testing dataset for validation structure is shown in section 7.1.

• Click on the **Start Training** button to begin model training.

• The **Stop Training** button allows users to halt the training process at any point.

• The **Open Folder Location** button opens the directory where training data and results are stored.

• The **Refresh** button clears all backend inputs and outputs, resetting the interface for a new training session.

8 Datasets Folder Structure

Proper organization of datasets is essential for effective model training and inference. The dataset structure follows a predefined format to ensure correct data handling.

8.1 Training Dataset

The **Training Dataset** must be structured in a classified format to allow the model to learn from labeled data. The dataset contains subfolders corresponding to individual kangaroos, each labeled as Roo_ID-N, where N ranges from 1 to 95, representing the fixed population of 95 individuals.

• Each subfolder Roo_ID-N contains images of the respective kangaroo n number of faces images.

• The dataset is divided into training and validation sets to prevent overfitting.

• When selecting the **Train Model** option, the user must browse the training dataset and the validation dataset using the **Browse Train Data** and **Browse Test Data** options, respectively.

• This structured format ensures the model is trained properly without bias.

The folder structure for the training dataset is as follows:

```
|-- Dataset/
| |-- Train Dataset/
| | |-- Roo_ID-1/
| | | |-- image_01.jpg
| | | |-- image_02.jpg
| | |-- Roo_ID-2/
| | |-- ...
| | |-- Roo_ID-95/
| |-- Test Dataset (Validation Data)/
| | |-- Roo_ID-1/
| | | |-- image_01.jpg
| | | |-- image_02.jpg
| | |-- Roo_ID-2/
| | |-- ...
| | |-- Roo_ID-95/
```

8.2 Test New Dataset

The **Test New Dataset** contains actual captured images of kangaroos in .jpg format. These images are used for inference once the model has been trained.

• The dataset includes raw images of kangaroos captured from different angles and

environments.

• The model processes these images to identify and classify the kangaroos.

• Users can select and browse these images using the **Test New Data** option in the interface.

• This dataset serves as real-world input to evaluate the trained model's performance.

The folder structure for the test dataset is as follows:

```
|-- Test_New_Dataset/
| |-- kangaroo_01.jpg
| |-- kangaroo_02.jpg
| |-- kangaroo_03.jpg
| |-- ...
```

9 Troubleshooting

If the software encounters issues such as not responding or crashing, follow these steps:

• Click the **Refresh** button to clear inputs and outputs and try again.

• Restart the system and try again.

• Relaunch the executable file.