# REAL-TIME TRAFFIC ANALYTICS

Documentation for “app.py”

## Overview

This document outlines the functionality of the Real-Time Traffic Analytics application, which processes video feeds to detect and count objects within a specified Region of Interest (ROI). The system uses a YOLO-based deep learning model for object detection and DeepSort for object tracking, integrated into a Streamlit interface for ease of use.

**NOTE** – Detailed line by line documentation is provided inline comments itself.

## Key Features

- **Camera Profile Management**: Users can select camera profiles that include predefined ROIs for accurate object detection.
- **Custom Object Detection**: Utilizes a custom-trained YOLO model to detect various objects within the video.
- **Real-Time Tracking and Counting**: Combines YOLO and DeepSort for tracking and counting objects within the ROI.
- **User Configurable Parameters**: Users can control the device (CPU/GPU), frame skip rate, and confidence threshold through the UI.
- **Streamlit Interface**: A web-based interface allows users to upload videos, select profiles, and visualize results.
-

## Installation & Setup

Prerequisites

- Python 3.x: Ensure Python 3.10 is installed.
- Required Libraries: The following Python libraries are required:
    - Streamlit
    - OpenCV
    - NumPy
    - Pandas
    - PyTorch
    - Ultralytics YOLO
    - DeepSort-RealTime

Installation Steps

1. Install Required Libraries: Install the necessary dependencies using pip:
2. “pip install streamlit opencv-python-headless numpy pandas torch ultralytics deep_sort_realtime”
3. Camera Profiles Setup: Ensure the camera_profiles.xlsx file is placed in the appropriate directory. This file should contain the camera profile IDs, locations, and ROI coordinates.
4. YOLO Model Weights: Place the YOLO model weights file in the specified path defined in the MODEL_WEIGHTS_PATH variable.
5. Temporary Video Directory: Create a directory named temp/ where uploaded videos will be temporarily stored during processing.

## Functionality Breakdown

1. Camera Profiles Management

- Camera Profiles Loading: Camera profiles are loaded from an Excel file containing details such as profile ID, location, and ROI coordinates.
- ROI Definition: The ROI for each camera profile is defined as a set of coordinates that form a polygon. These coordinates are used to determine whether detected objects fall within the specified area.

2. Video Processing

- Object Detection and Tracking:
    - The application loads the YOLO model for object detection.
    - DeepSort is used for tracking objects across frames, ensuring consistent counting even when objects move.
- Resizing with Padding: To maintain aspect ratio while resizing frames to the target size, padding is added to the images.
- ROI Filtering: Detected objects are filtered based on whether their center lies within the defined ROI.
- Counting Mechanism: Objects within the ROI are counted, with each unique track ID representing a distinct object.

3. Streamlit Interface

- Device Selection: Users can choose between CPU and CUDA (GPU) for processing.
- Profile Selection: Users select a camera profile, which determines the location and ROI to be used during video processing.
- Video Upload: Users upload videos directly through the interface. These videos are temporarily stored and then processed.

- ROI Visualization: The first frame of the uploaded video is displayed with the defined ROI overlaid, allowing users to visually confirm the accuracy of the ROI.
- Processing and Output: Once the video is processed, the application displays a summary of detected objects, categorized by type. The temporary video file is then deleted to save storage.

## Usage Instructions

1. Start the Application: Run the Streamlit app by navigating to the project directory and executing:
2. streamlit run app.py
3. 
4. Configure Settings:
    - Select the computation device (CPU or CUDA).
    - Choose a camera profile that matches the uploaded video.
5. Upload Video:
    - Upload a video file through the interface. Ensure the video format is supported (e.g., mp4, avi, mov, mkv).
6. Run Analysis:
    - Click the "Process Video" button to begin processing. The application will analyze the video, detect and track objects within the ROI, and display the count of each detected object.
7. Review Results:
    - View the results, including the number of each type of object detected within the ROI.
    - The uploaded video file is automatically deleted after processing.

## Error Handling & Logging

- **Error Handling**: The application includes basic error handling for issues such as file I/O errors or invalid video formats.
- **Logging**: Minimal logging is implemented for debugging purposes. Future enhancements could include more detailed logging for better traceability.