Defence Object Detection Project

Deva Vinoth

01/07/2024

Introduction:

In this project, we aim to detect various objects in a military video, including army personnel, weapons, aircraft, shoes, and tanks. We use the YOLOv5 model, a state-of-the-art object detection algorithm, to achieve this task.

Methodology:

- **YOLOv5 Model**

YOLOv5 (You Only Look Once version 5) is a real-time object detection model that is highly efficient and accurate. It divides the image into a grid and predicts bounding boxes and probabilities for each grid cell.

Implementation:

Step-by-Step Process

1. **Setup the Environment**: Install the required libraries and download the pre-trained YOLOv5 model.
2. **Load the Model**: Load the YOLOv5 model using PyTorch.
3. **Process the Video**: Use OpenCV to read the video frame by frame.
4. **Detect Objects**: Apply the YOLOv5 model to each frame to detect objects.
5. **Draw Bounding Boxes**: Draw bounding boxes and labels on the detected objects.
6. **Save the Output**: Write the processed frames to an output video file.

Python Code

```
import torch

import cv2

import time


# Loading the YOLOv5 model

model = torch.hub.load('ultralytics/yolov5', 'yolov5s')


# Function to process the video and detect objects

def detect_objects_in_video(video_path):

    # Open the video file

    cap = cv2.VideoCapture(video_path)
```

```
    # Get video properties
    width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
    fps = int(cap.get(cv2.CAP_PROP_FPS))

    # Define the codec and create VideoWriter object
    out = cv2.VideoWriter('output.mp4', cv2.VideoWriter_fourcc(*'XVID'), fps, (width, height))

    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break

        # Detect objects
        results = model(frame)

        # Convert results to pandas DataFrame
        df = results.pandas().xyxy[0]

        # Draw bounding boxes and labels on the frame
        for _, row in df.iterrows():
            x1, y1, x2, y2, conf, cls = int(row['xmin']), int(row['ymin']), int(row['xmax']),
int(row['ymax']), row['confidence'], row['name']

            # Replace labels as per requirements
```

```
            if cls == 'person':
                cls = 'soldier'
            elif cls == 'cow':  # in my code, I have replaced 'cow' with 'soldier' because, it predicting some soldiers as cow
                cls = 'soldier'
            elif cls == 'truck':
                cls = 'tanker'

            label = f'{cls} {conf:.2f}'
            cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)
            cv2.putText(frame, label, (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

        # Writing the frame into the output file
        out.write(frame)

        # Display the frame
        cv2.imshow('Frame', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
    out.release()
    cv2.destroyAllWindows()
```

```
# input video path

video_path = 'military-clips.mp4'


# Call the function to detect objects in the video

detect_objects_in_video(video_path)
```

Results:

The model successfully detected and labeled various objects in the input video. Below are some sample screenshots of the detection results.

Conclusion:

This project demonstrates the application of the YOLOv5 model for detecting military objects in a video. The results show that the model is capable of identifying and labeling multiple objects efficiently. Future work could involve training the model on a larger dataset specific to military objects to improve accuracy.

References:

1. YOLOv5: https://github.com/ultralytics/yolov5
2. OpenCV: https://opencv.org/