Instructions

Installation

Prerequisites

Python 3.8+
CUDA 11.0+ (optional, for GPU acceleration)

Setup

Install dependencies:
pip install -r requirements.txt

Training the detecter:

Train with dummy data (for testing):
python main.py train --dummy-data

Train with custom data:
python main.py train --config config.yaml

Inference

Detect objects in an image:
python main.py detect --model models/final_model.h5 --input-image test.jpg

Detect objects in a video:
python main.py detect --model models/final_model.h5 --input-video test.mp4 --output-video output.mp4

Real-time detection from webcam:
python main.py detect --model models/final_model.h5 --webcam --duration 30

Model Architecture:

Backbone: Transfer learning from pre-trained models (MobileNetV2, ResNet50, EfficientNet)
Detection Head: Custom layers for bounding box regression and classification
Outputs: Bounding boxes: [x1, y1, x2, y2]
Confidence score: Object presence probability
Class predictions: 80-class softmax output

Models used

TensorFlow 2.15+

NumPy

OpenCV

Pillow

Matplotlib

Scikit-learn

https://huggingface.co/spaces/Roboflow/Trackers/tree/main

Algorithm

It uses a custom data set to create any algorithm to detect what kind of debris that you may want to find and locate.

Example

```
from src.dataset import ObjectDetectionDataset

dataset = ObjectDetectionDataset(
   image_dir='data/train/images',
   annotation_dir='data/train/annotations',
   image_size=(416, 416),
   class_names=['person', 'car', 'dog']
)

train_gen = dataset.get_data_generator(batch_size=32, shuffle=True)
```

Technology stack

The system supports Pascal VOC XML format for annotations:

```
<?xml version="1.0"?>
<annotation>
  <object>
    <name>person</name>
    <bndbox>
      <xmin>100</xmin>
      <ymin>150</ymin>
      <xmax>300</xmax>
      <ymax>450</ymax>
    </bndbox>
  </object>
</annotation>
```