MINI PROJECT REPORT

Object Detection and Item Counting in Images Using YOLOv5

Abstract

Object detection is a pivotal aspect of computer vision, widely applied across industries like retail, healthcare, and autonomous driving. This project aims to implement a real-time object detection and item counting system using the YOLOv5 (You Only Look Once) model. YOLOv5, known for its speed and accuracy, is employed to detect and classify objects within an image. The project provides a user-friendly interface where users can upload images, and the model outputs the image with bounding boxes around detected objects. Additionally, the system counts and lists the number of objects detected. The integration of the PyTorch framework, along with supporting libraries such as Pillow, Matplotlib, and Ipywidgets, enables efficient and interactive processing. The project demonstrates the practical application of YOLOv5 in inventory management, real-time monitoring, and other object detection tasks.

Table of Contents

1. Introduction

Object detection is a fundamental task in computer vision that involves identifying instances of objects from a particular class within an image. Object detection models are widely used in a variety of applications, such as autonomous driving, video surveillance, inventory management, and healthcare.

This project focuses on developing a real-time object detection system using YOLOv5 (You Only Look Once, version 5) and counting objects within images. By integrating a web-based interface, users can upload images, and the system will detect, classify, and count the objects present in the image.

1.1 Objective

- To implement an object detection model using YOLOv5.
- To enable real-time detection of objects in images.
- To count and display the number of objects detected in the image.

1.2 Importance of Object Detection

Object detection is a critical task in computer vision that combines image classification and localization, allowing computers to detect and classify objects within an image or video. Its significance spans a wide range of fields and applications:

1. **Automation and Robotics**
   Object detection plays a crucial role in enabling machines to perceive and interact with their environment. Robots in manufacturing, for instance, can detect and manipulate objects, increasing efficiency and reducing human intervention.
2. **Autonomous Vehicles**
   In self-driving cars, object detection is fundamental for identifying pedestrians, vehicles, traffic signs, and obstacles in real-time, ensuring safety and making decisions regarding navigation.

2. Project Overview

2.1 Tools and Technologies Used

- **YOLOv5:**

  The core object detection model used in this project. YOLOv5 is known for its speed and accuracy, making it suitable for real-time applications.

- **Torch (PyTorch):**

  A deep learning library that facilitates the implementation and training of neural networks, used here to load and run the YOLOv5 model.

- **Matplotlib:**

  A plotting library used to display the images with detected bounding boxes.

- **Ipywidgets:**

  A library for creating interactive widgets, used for the image upload feature.

- **Pillow (PIL):**

  A Python library for image manipulation and processing.

- **Numpy:**

  A library for handling numerical computations and converting images into arrays.

3. Methodology

3.1 System Architecture

The system consists of three key components:

1. **User Interface:**

   A file upload widget for the user to upload images.

2. **YOLOv5 Model:**

   Pre-trained on COCO dataset and used to detect objects in the uploaded image.

3. **Object Detection Output:**

   Displays the uploaded image with bounding boxes drawn around the detected objects, along with a count of each object type.

3.2 Workflow

1. **Image Upload:**
   - Users upload an image through an interactive file upload widget built using ipywidgets. The widget accepts any image format (e.g., JPG, PNG).
2. **Image Processing:**
   - The uploaded image is converted to a suitable format using Pillow (PIL) and numpy.
   - The image is passed to the YOLOv5 model, which predicts bounding boxes, class labels, and confidence scores for each detected object.
3. **Object Detection:**
   - The YOLOv5 model processes the image and draws bounding boxes around each detected object.
   - Detected objects are categorized based on pre-trained labels (e.g., "person", "car", "bottle", etc.).
4. **Rendering Results:**

- The results (image with bounding boxes) are rendered using matplotlib.
- A summary of the detected objects is displayed, including the count of each item.

5. **Object Counting:**
   - The detected objects are output in a tabular format (using pandas), and their occurrences are counted to provide a summary of the items detected in the image.

4. Implementation

4.1 Code

```
import torch
from matplotlib import pyplot as plt
import ipywidgets as widgets
from IPython.display import display
from PIL import Image
import io
import numpy as np
import warnings
# Suppress future warnings
warnings.filterwarnings("ignore", category=FutureWarning)
# Load the pre-trained YOLOv5 model
model = torch.hub.load('ultralytics/yolov5', 'yolov5s', pretrained=True)
# Create a file upload widget for the user to upload their image
upload_widget = widgets.FileUpload(accept='image/*', multiple=False)
# Function to handle file upload and process the image
def handle_upload(change):
    # Get the uploaded file
    uploaded_file = list(change['new'].values())[0]
    # Convert the file content into a byte stream and read the image
    image_stream = io.BytesIO(uploaded_file['content'])
    image = Image.open(image_stream).convert('RGB')  # Ensure it's in RGB format
    # Convert the image into a format suitable for YOLOv5 model
```

```
    img_np = np.array(image)
    # Run the image through the YOLOv5 model for object detection
    results = model(img_np)
    # Render the results (draw bounding boxes on the image)
    rendered_img = np.squeeze(results.render())  # Render the detections on the image
    # Display the input image with bounding boxes drawn around detected objects
    plt.figure(figsize=(10, 10))
    plt.imshow(rendered_img)
    plt.title('Detected Objects in Cart')
    plt.axis('off')
    plt.show()
    # Count the number of each detected item
    detected_objects = results.pandas().xyxy[0]  # Get the results in pandas DataFrame format
    object_counts = detected_objects['name'].value_counts()  # Count occurrences of each
object
    # Display the counts of each object
    print("Item counts in the cart:")
    for item, count in object_counts.items():
        print(f"{item}: {count}")
# Bind the file upload widget to the handler function
upload_widget.observe(handle_upload, names='value')
# Display the widget
display(upload_widget)
```

4.2 Explanation of Key Components:

- **YOLOv5 Model:**

  Loaded using torch.hub and pre-trained on the COCO dataset. This model detects objects in the uploaded image.

- **File Upload Widget:**

  The ipywidgets.FileUpload() function allows users to upload images from their local system.

- **Image Processing:**

Images are processed using Pillow and numpy to convert them into a suitable format for the model.

- **Rendering and Display:**

The matplotlib library is used to display the uploaded image with detected objects. Bounding boxes and object labels are drawn around the detected objects.

- **Object Counting:**

After detection, the counts of each detected object are displayed to give users insight into the number of items identified.

4.3 OUTPUT

```
pip install torch matplotlib ipywidgets pillow
Requirement already satisfied: decorator in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (3.0.48
Requirement already satisfied: pygments in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (2.18.0)
Requirement already satisfied: backcall in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.2.0)
Requirement already satisfied: matplotlib-inline in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.1.7)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (4.9.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Requirement already satisfied: notebook>=4.4.1 in /usr/local/lib/python3.10/dist-packages (from widgetsnbextension~=3.6.0->ipywidgets) (6.5.5)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch) (3.0.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch) (1.3.0)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython>=4.0.0->ipywidgets) (0.8.4)
Requirement already satisfied: pyzmq<25,>=17 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (24.0.1)
Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (23.1.0)
Requirement already satisfied: jupyter-core>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (5.7
Requirement already satisfied: nbformat in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (5.10.4)
Requirement already satisfied: nbconvert>=5 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (6.5.4)
Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.6.0
Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.8.3
Requirement already satisfied: terminado>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (0.18.1
Requirement already satisfied: prometheus-client in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (0.21.
Requirement already satisfied: nbclassic>=0.4.7 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.1.0)
Requirement already satisfied: ptyprocess>=0.5 in /usr/local/lib/python3.10/dist-packages (from pexpect>4.3->ipython>=4.0.0->ipywidgets) (0.7.0)
```

```
pip install torch torchvision torchaudio matplotlib ipywidgets pillow
Requirement already satisfied: jupyter-client in /usr/local/lib/python3.10/dist-packages (from ipykernel>=4.5.1->ipywidgets) (6.1.12)
Requirement already satisfied: tornado>=4.2 in /usr/local/lib/python3.10/dist-packages (from ipykernel>=4.5.1->ipywidgets) (6.3.3)
Requirement already satisfied: setuptools>=18.5 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (71.0.4)
Requirement already satisfied: jedi>=0.16 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.19.1)
Requirement already satisfied: decorator in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (4.4.2)
Requirement already satisfied: pickleshare in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.7.5)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (3.0.48)
Requirement already satisfied: pygments in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (2.18.0)
Requirement already satisfied: backcall in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.2.0)
Requirement already satisfied: matplotlib-inline in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (0.1.7)
Requirement already satisfied: pexpect>4.3 in /usr/local/lib/python3.10/dist-packages (from ipython>=4.0.0->ipywidgets) (4.9.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
Requirement already satisfied: notebook>=4.4.1 in /usr/local/lib/python3.10/dist-packages (from widgetsnbextension~=3.6.0->ipywidgets) (6.5.5)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch) (3.0.1)
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch) (1.3.0)
Requirement already satisfied: parso<0.9.0,>=0.8.3 in /usr/local/lib/python3.10/dist-packages (from jedi>=0.16->ipython>=4.0.0->ipywidgets) (0.8.4)
Requirement already satisfied: pyzmq<25,>=17 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (24.0.1)
Requirement already satisfied: argon2-cffi in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (23.1.0)
Requirement already satisfied: jupyter-core>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (5.7.2)
Requirement already satisfied: nbformat in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (5.10.4)
Requirement already satisfied: nbconvert>=5 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (6.5.4)
Requirement already satisfied: nest-asyncio>=1.5 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.6.0)
Requirement already satisfied: Send2Trash>=1.8.0 in /usr/local/lib/python3.10/dist-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets) (1.8.3)
```

Input images :

Output :

```
Using cache found in /root/.cache/torch/hub/ultralytics_yolov5_master
YOLOv5 🚀 2024-10-17 Python-3.10.12 torch-2.4.1+cu121 CPU

Fusing layers...
YOLOv5s summary: 213 layers, 7225885 parameters, 0 gradients, 16.4 GFLOPs
Adding AutoShape...
     Upload (1)
Item counts in the cart:
bottle: 5
apple: 3
broccoli: 1
```

5.Conclusion

In conclusion, this project successfully implemented an object detection and counting system using the YOLOv5 model, providing accurate and real-time results. The integration of PyTorch, along with supporting libraries like Matplotlib and Ipywidgets, allowed for efficient image processing and a user-friendly interface. The system can detect and classify multiple objects within an image while also counting them. This demonstrates the potential for practical applications such as inventory management and surveillance. Future enhancements could include improving detection speed and expanding the system to handle video streams.

6. Future Improvements

- Integrating video processing capabilities for continuous detection.
- Adding support for custom datasets to detect domain-specific objects.
- Implementing real-time notifications for specific object detections

7.References

- **YOLOv5 GitHub Repository**: https://github.com/ultralytics/yolov5
- **Torch Documentation**: https://pytorch.org/docs/
- **Ipywidgets Documentation**: https://ipywidgets.readthedocs.io/