使用YOLOv8和ByteTracker进行实时人员跟踪和计数

Original 二旺 小白玩转Python 2024年08月25日 20:01 福建

在计算机视觉领域，实时跟踪和统计人数对于各种应用至关重要，从监控到事件管理。在这篇博客文章中，我们将探讨如何利用YOLOv8和ByteTracker实现准确的人数统计。

引言

1. YOLOv8（You Only Look Once，第八版）是一种以其速度和准确性而闻名的最新对象检测模型。
2. ByteTracker是一种先进的跟踪算法，旨在维持对象在帧之间的身份，使其成为执行人数统计等任务的理想选择。

这种组合不仅允许我们在帧中检测到人，而且还能够跟踪他们在帧之间的移动，为实时人数统计提供了强大的解决方案。

先决条件

在深入实现之前，请确保您具备以下条件：

- Python 3.10
- Ultralytics

设置环境

首先，创建并激活虚拟环境以管理依赖项：

```
conda create -n person-tracker python==3.10
conda activate person-tracker
```

安装必要的库：

```
pip install ultralytics
```

实现人数统计

1. 导入库从导入所需的库开始：

```
from ultralytics import YOLO
from datetime import datetime
import os
```

2. 定义PersonTracker类

创建一个PersonTracker类，该类集成了用于检测的YOLOv8和用于跟踪的ByteTracker：

```
class PersonTracker:
    def __init__(self, model_path, result_dir='results/', tracker_config="byt
                 iou=0.5, img_size=(720, 1080)):
        self.model = YOLO(model_path)
        self.result_dir = result_dir
        self.tracker_config = tracker_config
        self.conf = conf
        self.device = device
        self.iou = iou
        self.img_size = img_size

    def create_result_file(self):
        folder_name = datetime.now().strftime("%Y-%m-%d-%H-%M-%S")
        result_file_path = os.path.join(self.result_dir, folder_name + ".txt"
        os.makedirs(self.result_dir, exist_ok=True)
        with open(result_file_path, 'w') as file:
            file.write(folder_name + "\n")
        return result_file_path
```

```
    def detect_and_track(self, source, show=True, logger=None):
        result_file = self.create_result_file()
        person_count = 0
        previous_person_count = 0

        results = self.model.track(
            source, show=show, stream=True, tracker=self.tracker_config, conf
            device=self.device, iou=self.iou, stream_buffer=True, classes=[0]
        )

        for i, result in enumerate(results):
            boxes = result.boxes
            try:
                id_count = boxes.id.int().tolist()
                max_id = max(id_count)

                if max_id > person_count:
                    person_count = max_id

                if person_count != previous_person_count:
                    previous_person_count = person_count
                    with open(result_file, 'a') as filewrite:
                        filewrite.write(f"Person count: {person_count}\n")

                    if logger:
                        logger.info(f"Person count: {person_count}")

            except Exception as e:
                pass
```

3. 运行人员跟踪器

- 视频文件跟踪

```
if __name__ == '__main__':
    source = "path/to/your/video.mp4"
    tracker = PersonTracker(model_path='path/to/yolov8_model.pt')
    tracker.detect_and_track(source=source)
```

- 网络摄像头跟踪

```
if __name__ == '__main__':
    source = 0  # Use 0 for the default webcam
    tracker = PersonTracker(model_path='path/to/yolov8_model.pt')
```

```
    tracker.detect_and_track(source=source)
```

- RTSP流跟踪

```
if __name__ == '__main__':
    source = 'rtsp://username:password@ip_address:port/path'
    tracker = PersonTracker(model_path='path/to/yolov8_model.pt')
    tracker.detect_and_track(source=source)
```

支持的模型

对于YOLOv8，模型通常根据它们的准确性和速度权衡进行分类。通常支持以下模型：

- YOLOv8n（Nano）：提供高速度和较低的准确性。非常适合处理速度的实时应用
- YOLOv8s（Small）：平衡速度和准确性。适用于许多实际应用。
- YOLOv8m（Medium）：在速度和准确性之间提供良好的权衡。适用于更苛刻的应用。
- YOLOv8l（Large）：高准确性，速度较低。最适合准确性为优先考虑的场景。

结论

通过结合YOLOv8和ByteTracker，您可以有效地在帧之间检测和跟踪人员，提供准确的计数和有价值的洞察。这个解决方案可以扩展到需要实时个人监控和分析的各种应用。

参考文献

- YOLOv8文档：https://docs.ultralytics.com/
- ByteTracker论文：https://arxiv.org/abs/2005.03659
- 源码：https://github.com/ishworrsubedii/person_tracker

• END •