ĐẠI HỌC HUẾ
TRƯỜNG ĐẠI HỌC KHOA HỌC

BẢN THUYẾT MINH SẢN PHẨM DỰ THI

CUỘC THI LẬP TRÌNH DÀNH CHO HỌC SINH TRUNG HỌC PHỔ THÔNG VÀ SẢN PHẨM SÁNG TẠO CÔNG NGHỆ THÔNG TIN DÀNH CHO SINH VIÊN CAO ĐẲNG, ĐẠI HỌC NĂM 2024 (HUE-ICT CHALLENGE-2024)

Tên sản phẩm:

XÂY DỰNG ỨNG DỤNG TỰ ĐỘNG PHÁT HIỆN PHƯƠNG TIỆN VI PHẠM HIỆU LỆNH ĐÈN TÍN HIỆU GIAO THÔNG

Lĩnh vực: **TRÍ TUỆ NHÂN TẠO VÀ PHẦN MỀM**

Tác giả:

1. LÊ BÁ TUẤN PHONG

Thừa Thiên Huế, ngày 08 tháng 06 năm 2024

I NỘI DUNG

1 Tên sản phẩm

Ứng dụng tự động phát hiện phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông.

2 Ý tưởng của đề tài

2.1 Giới thiệu chung về đề tài

Trong bối cảnh giao thông đô thị ngày càng phức tạp, việc phát hiện và xử lý các phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông trở nên cực kỳ quan trọng nhằm giảm thiểu tai nạn và đảm bảo an toàn giao thông. Việc vi phạm hiệu lệnh đèn tín hiệu không chỉ gây ra nguy cơ tai nạn mà còn làm tăng tình trạng ùn tắc giao thông, ảnh hưởng đến cuộc sống hàng ngày của người dân. Để giải quyết vấn đề này, nhiều công nghệ tiên tiến đã được nghiên cứu và áp dụng, trong đó có sự phát triển mạnh mẽ của các thuật toán học sâu và trí tuệ nhân tạo. Đề tài "Xây dựng ứng dụng phát hiện phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông"tập trung vào việc sử dụng các công nghệ tiên tiến như YOLOv8, Paddle OCR và DeepSORT để xây dựng một hệ thống tự động phát hiện và xử lý vi phạm giao thông.

2.2 Lý do chọn đề tài

Việc phát hiện vi phạm giao thông hiện nay chủ yếu dựa vào sự quan sát của con người hoặc các hệ thống giám sát truyền thống, vốn có nhiều hạn chế về độ chính xác và khả năng xử lý thời gian thực. Trong khi đó, sự phát triển của các thuật toán học sâu và trí tuệ nhân tạo đã mở ra những cơ hội mới cho việc xây dựng các hệ thống giám sát giao thông thông minh, hiệu quả và chính xác

hơn. YOLOv8, một trong những phiên bản mới nhất của YOLO, đã chứng minh được khả năng vượt trội trong việc phát hiện đối tượng với độ chính xác cao và tốc độ xử lý nhanh. Bên cạnh đó, công nghệ nhận diện ký tự quang học (OCR) như Paddle OCR cũng đã đạt được những tiến bộ đáng kể, cho phép nhận diện biển số xe một cách chính xác trong nhiều điều kiện khác nhau. Việc tích hợp DeepSORT để theo dõi phương tiện càng làm tăng tính hiệu quả của hệ thống. Đề tài này không chỉ giúp cải thiện khả năng phát hiện vi phạm mà còn đóng góp vào nghiên cứu và ứng dụng các công nghệ mới trong lĩnh vực giao thông, đáp ứng nhu cầu cấp thiết của xã hội hiện đại trong việc đảm bảo an toàn giao thông và giảm thiểu tai nạn.

2.3 Thách thức

Các thách thức chính của đề tài bao gồm:

- **Độ chính xác và hiệu suất của mô hình YOLOv8**: Việc đảm bảo mô hình YOLOv8 hoạt động với độ chính xác cao trong việc phát hiện phương tiện và đèn tín hiệu giao thông là một thách thức lớn. Điều này đòi hỏi quá trình huấn luyện mô hình phải sử dụng một lượng dữ liệu lớn và đa dạng, đồng thời tối ưu hóa các tham số của mô hình.
- **Khả năng nhận diện biển số xe trong điều kiện ánh sáng và thời tiết khác nhau**: Điều kiện ánh sáng yếu, thời tiết xấu như mưa, sương mù có thể ảnh hưởng đến khả năng nhận diện biển số xe của hệ thống. Do đó, cần phải có các biện pháp xử lý hình ảnh tiên tiến để cải thiện độ chính xác trong các điều kiện này.
- **Tích hợp các thành phần khác nhau để tạo thành một hệ thống hoàn chỉnh**: Việc tích hợp các thành phần như YOLOv8, Paddle OCR và Deep-

SORT để tạo thành một hệ thống hoạt động mượt mà và hiệu quả là một thách thức lớn. Điều này đòi hỏi kiến thức sâu sắc về từng thành phần cũng như khả năng lập trình và tích hợp hệ thống.

- **Đảm bảo hệ thống hoạt động thời gian thực**: Việc xử lý và phân tích video thời gian thực đòi hỏi hệ thống phải có hiệu suất cao và khả năng tối ưu hóa tài nguyên tính toán. Điều này đòi hỏi các thuật toán phải được tối ưu hóa và hệ thống phần cứng phải đủ mạnh để đáp ứng yêu cầu.
- **Bảo mật và bảo vệ dữ liệu**: Việc thu thập và xử lý dữ liệu giao thông đòi hỏi phải đảm bảo tính bảo mật và bảo vệ dữ liệu cá nhân. Điều này đòi hỏi các biện pháp bảo mật mạnh mẽ và tuân thủ các quy định về bảo vệ dữ liệu.

II MÔ TẢ VỀ SẢN PHẨM

1 Tính mới, tính sáng tạo của sản phẩm

Sản phẩm sử dụng ba công nghệ tiên tiến: YOLOv8, Paddle OCR, và DeepSORT, để xây dựng một hệ thống tự động phát hiện và xử lý vi phạm giao thông.

- **YOLOv8** [1]: Đây là một trong những phiên bản mới nhất của mô hình YOLO, nổi bật với độ chính xác cao và tốc độ xử lý nhanh, giúp phát hiện đối tượng một cách hiệu quả
- **Paddle OCR** [2]: Sử dụng công nghệ nhận diện ký tự quang học tiên tiến để nhận diện biển số xe, đảm bảo độ chính xác cao trong nhiều điều kiện khác nhau.
- **DeepSORT** [3]: Công nghệ theo dõi đối tượng giúp duy trì tính liên tục và chính xác trong việc gán biển số xe cho từng phương tiện.
- **Tính sáng tạo**: Tích hợp ba công nghệ này vào một hệ thống duy nhất giúp cải thiện đáng kể độ chính xác và hiệu suất trong việc giám sát giao thông thời gian thực. Hệ thống có khả năng xử lý và phân tích video với hiệu suất cao, đảm bảo hoạt động hiệu quả trong các tình huống thực tế phức tạp .

2 Các nguyên vật liệu làm ra mô hình, sản phẩm

Việc xây dựng hệ thống tự động phát hiện phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông yêu cầu sử dụng nhiều nguyên vật liệu và công cụ phần cứng lẫn phần mềm. Dưới đây là chi tiết về các thành phần cần thiết để làm ra mô hình này:

2.1 Phần cứng

2.1.1 Máy tính hoặc Máy chủ

- **CPU**: Một bộ xử lý mạnh mẽ để đảm bảo tốc độ xử lý nhanh chóng cho các tác vụ tính toán phức tạp.
- **GPU**: Một card đồ họa mạnh mẽ để tăng tốc quá trình huấn luyện và phân tích video, đặc biệt là các mô hình học sâu yêu cầu hiệu suất cao như YOLOv8.
- **RAM**: Ít nhất 8GB RAM để đảm bảo khả năng xử lý mượt mà các tác vụ liên quan đến hình ảnh và video.

2.2 Phần mềm

2.2.1 Hệ điều hành và Môi trường phát triển

- **Hệ điều hành**: Ubuntu Linux hoặc Windows, phù hợp với các thư viện và công cụ phát triển học sâu.
- **Python**: Ngôn ngữ lập trình chính để phát triển các mô hình và xử lý dữ liệu.
- **Anaconda**: Quản lý môi trường phát triển và các thư viện cần thiết trong pytpython.

2.2.2 Thư viện và Công cụ học sâu

- **YOLOv8**: Phiên bản mới nhất của YOLO, được phát triển bởi Ultralytics, sử dụng để phát hiện các đối tượng trong video.
- **Paddle OCR**: Thư viện mã nguồn mở được phát triển bởi PaddlePaddle, sử dụng để nhận diện ký tự quang học từ biển số xe.

- **DeepSORT**: Thuật toán theo dõi đối tượng thời gian thực, giúp duy trì tính liên tục và chính xác trong việc theo dõi các phương tiện.

2.2.3 Thư viện và Công cụ hỗ trợ khác

- **OpenCV**: Thư viện xử lý ảnh mở rộng, hỗ trợ việc xử lý và phân tích video.
- **NumPy và Pandas**: Thư viện xử lý dữ liệu, hỗ trợ trong việc quản lý và phân tích dữ liệu hình ảnh và video.
- **Tkinter**: Thư viện để phát triển giao diện người dùng (GUI), giúp hiển thị thông tin vi phạm và tương tác với người dùng.

2.3 Dữ liệu

- **Dữ liệu hình ảnh và video**: Tập dữ liệu giao thông bao gồm hơn 5000 hình ảnh của các phương tiện giao thông và đèn tín hiệu giao thông. Dữ liệu này được thu thập từ nguồn camera giám sát của Trung tâm giám sát và điều hành đô thị Thừa Thiên Huế.
- **Dữ liệu nhãn**: Các nhãn dữ liệu cho xe, biển số xe và đèn tín hiệu giao thông do tự bản thân gán nhãn, được sử dụng để huấn luyện và đánh giá các mô hình.

Việc sử dụng các nguyên vật liệu và công cụ này đảm bảo rằng hệ thống tự động phát hiện phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông được xây dựng với chất lượng cao và hiệu quả tốt nhất, đáp ứng được các yêu cầu kỹ thuật và vận hành trong thực tế.

3 Cách lắp ráp và cài đặt sản phẩm

Quá trình lắp ráp và cài đặt hệ thống tự động phát hiện phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông bao gồm các bước sau:

3.1 Chuẩn bị phần cứng

- **Máy tính hoặc Máy chủ**:
  - Lắp đặt máy tính hoặc máy chủ với CPU mạnh mẽ và GPU hiệu suất cao.
  - Gắn ít nhất 8GB RAM.
- **Camera Giám sát**:
  - Mở video hoặc thiết lập kết nối với luồng phát trực tiếp từ camera giám sát giao thông. Đảm bảo video có độ phân giải và chất lượng đủ tốt để phát hiện và nhận diện các phương tiện.

3.2 Chuẩn bị phần mềm

- **Hệ điều hành và Môi trường phát triển**:
  - Cài đặt hệ điều hành Ubuntu Linux hoặc Windows trên máy tính hoặc máy chủ.
  - Cài đặt Python và Anaconda để quản lý môi trường phát triển và các thư viện cần thiết.

3.3 Cài đặt sản phẩm

1. **Mở command Prompt và nhập: (clone kho lưu trữ)**:

```
git clone https://github.com/Bao-Tap/RedLightViolation.git
cd RedLightViolation
```

2. **Cài đặt các gói cần thiết**:

```
pip install -r requirements.txt
```

3.4 Sử dụng sản phẩm

```
python app.py
```

Quá trình lắp ráp và cài đặt sản phẩm đòi hỏi sự chuẩn bị kỹ lưỡng về phần cứng và phần mềm để đảm bảo hệ thống hoạt động hiệu quả và chính xác.

4 Nguyên tắc hoạt động, vận hành của sản phẩm dự thi

4.1 Nguyên tắc hoạt động:

- Phát hiện và nhận diện đối tượng: Sử dụng YOLOv8 để phát hiện vị trí của các phương tiện và đèn tín hiệu giao thông trong video.
- Nhận diện biển số xe: Sử dụng Paddle OCR để nhận diện các ký tự trên biển số xe từ các vùng được cắt ra bởi YOLOv8.
- Theo dõi đối tượng: Sử dụng DeepSORT để theo dõi các phương tiện qua các khung hình, duy trì tính liên tục của biển số xe.
- Xử lý hậu kỳ: Kiểm tra và chuẩn hóa kết quả nhận diện biển số xe, đảm bảo tuân thủ các quy định pháp lý hiện hành.
- Xuất kết quả: Lưu trữ và hiển thị kết quả vi phạm giao thông, cung cấp công cụ phân tích dữ liệu cho quản lý giao thông.

4.2 Quá trình hoạt động

4.2.1 Xác định vạch kẻ

Sau khi mở video hoặc luồng phát trực tiếp, để phát hiện các phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông, bước đầu tiên là xác định vị trí của các vạch kẻ trên đường. Các vạch kẻ này sẽ được sử dụng làm ranh giới để kiểm tra xem phương tiện có vượt qua khi đèn đỏ hay không hoặc có rẽ phải hay không. Người giám sát chỉ cần thao tác trên giao diện là có thể kẻ được các vạch kẻ này. (xem hình 1)

Hình 1: Minh họa vạch kẻ màu đỏ của hệ thống do người dùng tự vẽ.

4.2.2 Quá trình chi tiết phát hiện vi phạm

Sau khi xác định được vị trí của các vạch kẻ, chúng ta tiến hành phát hiện các phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông. Quá trình này bao

gồm các bước sau:

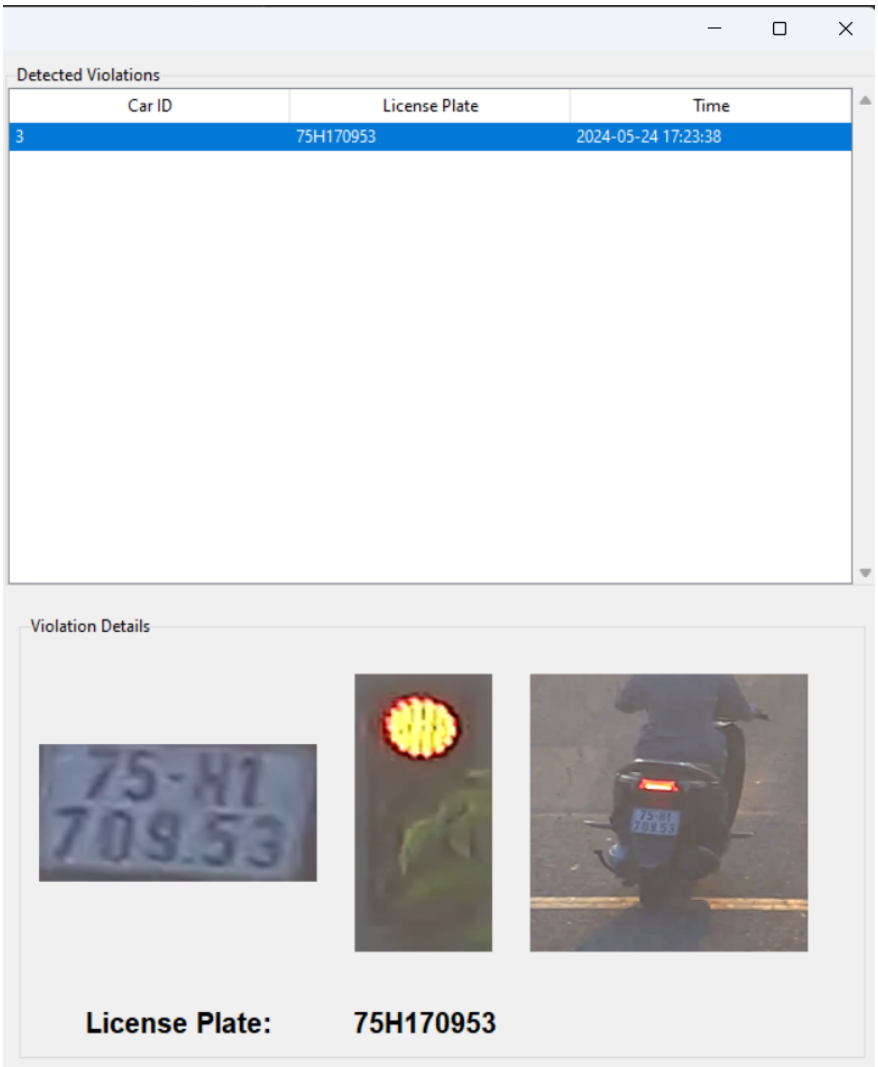
1. **Phát hiện phương tiện**: Sử dụng mô hình YOLOv8 để phát hiện các phương tiện trong khung hình.
2. **Phát hiện biển số xe và đọc biển số với Paddle OCR**: với mỗi biển số ban đầu xác định được thì ngay lập tức gán nó cho phương tiện sở hữu nó. Điều kiện gán là biển số phải hoàn toàn nằm trong bounding box của xe. (Trường hợp nếu 1 biển số nằm trong vùng của nhiều xe thì tiến hành gán cho xe có chỉ số IoU cao nhất)
3. **Theo dõi phương tiện**: Sử dụng DeepSORT để theo dõi các phương tiện ngay sau khi đọc được biển số xe của chúng. Mỗi chiếc xe được theo dõi sẽ bao gồm các thuộc tính: *biển số xe và độ tin cậy của biển số đó* (Nếu trường hợp ở các khung hình sau, biển số xe đọc được với độ tin cậy cao hơn lập tức thay đổi biển số xe mới), *trạng thái vi phạm*, *hướng di chuyển*, *vượt qua vạch hay chưa*.
4. **Xác định trạng thái đèn tín hiệu**: Xác định trạng thái của đèn tín hiệu giao thông (đỏ, không phải đỏ) tại mỗi thời điểm. Nếu sau 3 khung hình không phát hiện được đèn đỏ thì mặc định chuyển sang đèn xanh. Điều này làm giảm khả năng dự đoán sai của mô hình trong trường hợp bỏ xót 1 hoặc 2 khung hình không phát hiện được do nhiễu.
5. **Kiểm tra vị trí phương tiện**: So sánh vị trí của các phương tiện với vạch kẻ. Nếu hướng di chuyển của phương tiện là "tiến"và "chưa vượt qua vạch kẻ"thì khi vượt qua vạch kẻ khi đèn đỏ, lập tức ghi nhận thông tin vi phạm. Những trường hợp ngược lại không ghi nhận. (Những phương tiện được gọi là "tiến": khi lần đầu tiên xuất hiện nằm bên dưới vạch kẻ.)

6. **Trường hợp cho phép rẽ phải**: So sánh vị trí của các phương tiện với vạch kẻ rẽ phải. Nếu phương tiện đi vào vùng rẽ phải và phương tiện đang bị ghi nhận là vi phạm thì xóa thông tin vi phạm của phương tiện.

4.2.3 Hiển thị thông tin lên giao diện

Thông tin về các vi phạm sẽ được hiển thị lên giao diện người dùng để dễ dàng theo dõi và xử lý. Giao diện này được xây dựng bằng Python Tkinter và bao gồm các thành phần sau:

- **Hiển thị khung hình từ video, camera giám sát**: Hiển thị các khung hình của video giám sát, bao gồm các bounding box của các phương tiện và biển số xe đã được gán của phương tiện đó.



Hình 2: Minh họa giao diện của hệ thống khi chọn vào phương tiện vi phạm.

- **Thông tin vi phạm**: Hiển thị thông tin chi tiết về các phương tiện vi phạm, bao gồm thời gian, biển số, hình ảnh vi phạm. (xem hình 2)

5 Khả năng ứng dụng của sản phẩm

Hệ thống phát hiện vi phạm hiệu lệnh đèn tín hiệu giao thông có thể được ứng dụng trong nhiều lĩnh vực khác nhau, bao gồm:

- **Hệ thống giám sát giao thông tự động**: Tại các ngã tư, giao lộ, hệ thống có thể tự động giám sát và phát hiện các phương tiện vi phạm hiệu lệnh đèn tín hiệu giao thông. Điều này giúp giảm bớt gánh nặng cho lực lượng cảnh sát giao thông và tăng cường hiệu quả giám sát.

Hình 3: Một chú cảnh sát giao thông đang trực camera tại trung tâm chỉ huy.

- **Hỗ trợ lực lượng cảnh sát giao thông**: Hệ thống cung cấp thông tin chi tiết về các phương tiện vi phạm, bao gồm biển số xe, thời gian và vị trí vi phạm, giúp cảnh sát giao thông có thể xử lý vi phạm một cách nhanh chóng

và chính xác hơn.

- **Nghiên cứu và phát triển các hệ thống giao thông thông minh**: Các dữ liệu thu thập được từ hệ thống có thể được sử dụng cho các nghiên cứu về hành vi giao thông, đánh giá hiệu quả của các biện pháp quản lý giao thông và phát triển các giải pháp giao thông thông minh hơn.
- **Quản lý giám sát và điều hành giao thông đô thị**: Hệ thống giúp các cơ quan quản lý giao thông có cái nhìn tổng quan về tình hình giao thông tại các khu vực trọng điểm, từ đó đưa ra các biện pháp điều hành và quản lý giao thông hiệu quả hơn.

Hình 4: Hình ảnh tại Trung tâm giám sát điều hành đô thị tỉnh Thừa Thiên Huế.

- **Ứng dụng trong các khu vực tư nhân**: Các khu vực tư nhân như bãi đỗ xe, khu công nghiệp, khu dân cư có thể sử dụng hệ thống để giám sát và quản lý giao thông nội bộ, đảm bảo an toàn và trật tự giao thông.

6 Hiệu quả đạt được của sản phẩm

Trong quá trình thử nghiệm, hệ thống đã sử dụng 60 video vi phạm giao thông, mỗi video dài 30 giây. Kết quả thử nghiệm như sau:

- Tổng số đối tượng phương tiện vượt qua vạch kẻ: 232.
- Trong đó có 103 phương tiện vượt đèn đỏ, số vi phạm phát hiện được là 89, số biển số đọc đúng là 67.
- 71 trường hợp sau khi vượt đèn đỏ đã rẽ phải, hệ thống phát hiện được 29 trường hợp.
- Tốc độ khung hình trên CPU là 3 khung hình trên 1 giây.

Dựa trên các kết quả này, các chỉ số đánh giá hiệu suất của hệ thống được tính toán như sau dựa trên trường hợp không tính đến việc cho phép rẽ phải:

- **Độ chính xác (Accuracy)**:

$$\text{Accuracy} = \frac{\text{Số vi phạm phát hiện được}}{\text{Số vi phạm thực tế}} \times 100 = \frac{89}{103} \times 100 = 86.4\%$$

- **Độ chính xác (Precision)**:

$$\text{Precision} = \frac{\text{Số vi phạm phát hiện đúng}}{\text{Số vi phạm phát hiện được}} \times 100 = \frac{89}{89} \times 100 = 100\%$$

- **Độ nhạy (Recall)**:

$$\text{Recall} = \frac{\text{Số vi phạm nhìn thấy}}{\text{Tổng số vi phạm thực tế}} \times 100 = \frac{89}{103} \times 100 = 86.4\%$$

Như vậy, hệ thống đã đạt được các chỉ số đáng kể về độ chính xác, độ nhạy trong việc phát hiện các phương tiện vi phạm giao thông. Điều này cho thấy tiềm năng ứng dụng của hệ thống trong thực tế là rất cao, giúp hỗ trợ hiệu quả trong việc giám sát và xử lý vi phạm giao thông.

7 Địa chỉ đăng tải sản phẩm

Đường dẫn Github của ứng dụng: https://github.com/Bao-Tap/RedLightViolation

8 Cam kết về bản quyền sản phẩm

- Sản phẩm chưa từng được công bố hoặc tham gia trong bất kỳ cuộc thi nào trước đây;

- Sản phẩm đúng bản quyền của sinh viên dự thi, tuân thủ các yêu cầu giấy phép mã nguồn mở của các tổ chức, cá nhân phát hành mã nguồn mở.

Tài liệu tham khảo

[1] Ultralytics. Yolov8 documentation. `https://docs.ultralytics.com/`, 2024. Accessed: 2024-06-10.

[2] Yuning Duan, Chuning Cui, Xiang Xiao, Yu Wang, Guoqiang Chen, Shancheng Zhang, Shouhong Pu, Erjin Zhang, Chenxia Li, Pengfei Sun, Feiyu Lin, Dongchao Han, Yi Liu, and Xun Qiao. Pp-ocr: A practical ultra lightweight ocr system. *arXiv preprint arXiv:2009.09941*, 2020.

[3] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and real-time tracking with a deep association metric. *arXiv:1703.07402*, 2017.