

Báo cáo hàng tuần

Detect License Plate

by Cường x Huy

Ha Noi

03/11/2023

# Giới thiệu

Detect License Plate (DLP) là một hệ thống tiên tiến được thiết kế đặc biệt để theo dõi lưu lượng xe đi vào và đi ra nhằm mục đích giám sát lưu lượng xe. DLP sử dụng công nghệ nhận diện biển số xe hiệu quả bằng mô hình Yolo V5 để tự động ghi lại thông tin về các phương tiện tham gia giao thông như realtime, biển số….

# Quy trình

TUẦN 1 : 27/11 - 2/12

Flask lại thẻ, cài hệ điều hành cho con Jetson nano.

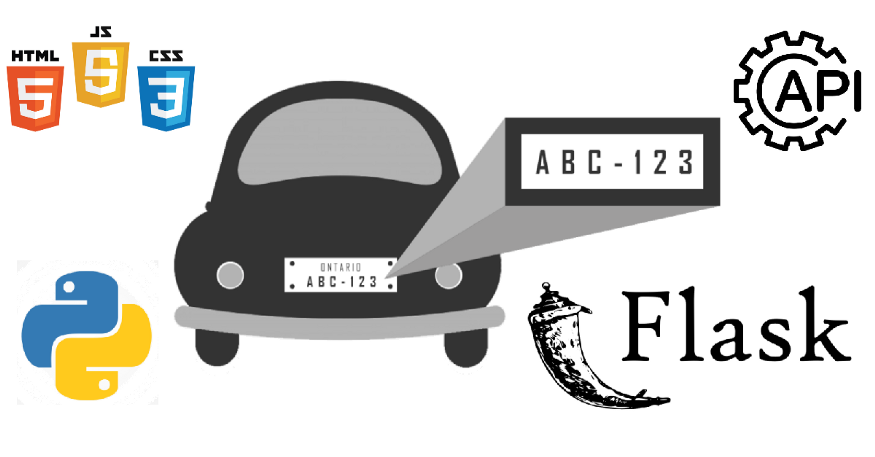
Run deploy.py trên máy tính cá nhân.

Cài các thư viện riêng cho Jetson nano.

Luyện lại model bằng mô hình yolo v5.

Chuyển thành công từ file .engine sang file .pt (thành công lớn).

## 

****

**1**. **Run deploy.py trên máy tính cá nhân**

git clone project : <https://github.com/Jackson2706/Detect-license-plate.git>

Cài các thư viện cần thiết

Cài lại thư viện Tesseract qua link(với windown): https://github.com/UB-Mannheim/tesseract/wiki

- Với linux: chỉ cần lệnh:  
sudo apt update

sudo apt install tesseract-ocr

**2. Cài các thư viện riêng cho Jetson nano**

Numpy:1.13.3

Pytorch: Cài theo docs [NVIDIA Jetson Nano Deployment - Ultralytics YOLOv8 Docs](https://docs.ultralytics.com/yolov5/tutorials/running_on_jetson_nano/#install-pytorch-and-torchvision) trên github [ultralytics/yolov5: YOLOv5 🚀 in PyTorch > ONNX > CoreML > TFLite (github.com)](https://github.com/ultralytics/yolov5) bản 1.10.0 và torchvision 0.11.1

Opencv:4.1.1 cài theo vid [(1623) Install Opencv 4.1 on Nvidia Jetson Nano - YouTube](https://www.youtube.com/watch?v=SSzZZ68jy_I&t=517s)

Các thư viện khác cài theo lệnh **sudo python3 -m pip install** **(tên thư viện==phiên bản-nếu có)**

## Doc cài jetsion nano

**B1:FLASH THẺ NHỚ CHO JETSION NANO**

- Download OS cho jetsion nano theo đường dẫn [JetPack SDK 4.6.1 | NVIDIA Developer](https://developer.nvidia.com/embedded/jetpack-sdk-461)

về máy tính cá nhân

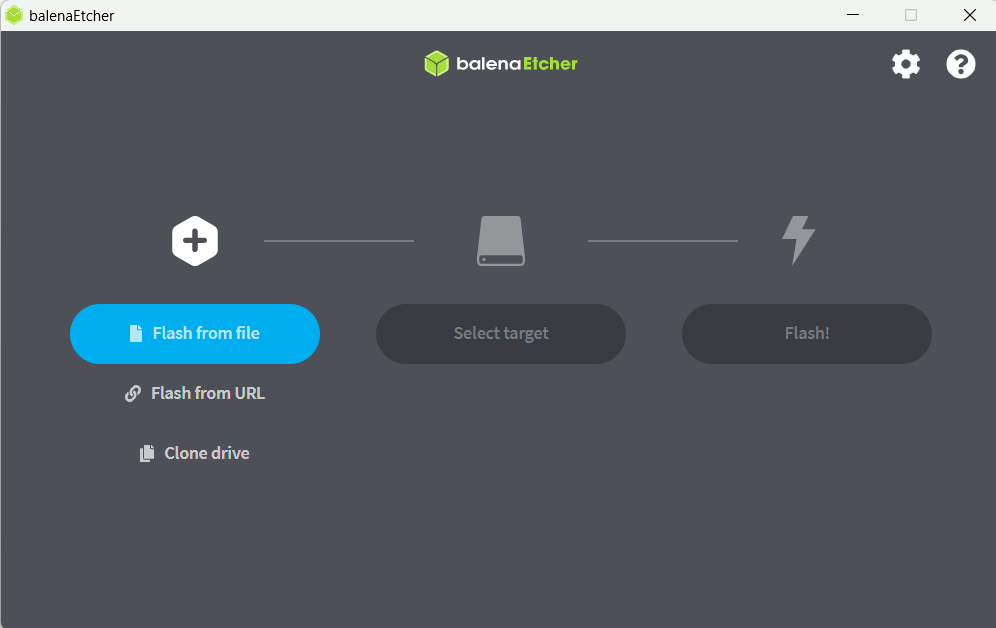
- Download [balenaEtcher - Flash OS images to SD cards & USB drives](https://etcher.balena.io/) để flash

-Bắt đầu flash :

+ Cắm thẻ nhớ vào đầu đọc thẻ trên lab



+ Cắm vào máy tính cá nhân , vào balenaEtcher để flash





+ Chọn bản .zip của hệ điều hành sau đó bắt đầu flash vào thẻ

- Sau khi flash cắm thẻ vào jetsion và khởi động

**B2 : CÀI THƯ VIỆN**

Numpy:1.13.3

Pytorch:Cài theo docs [NVIDIA Jetson Nano Deployment - Ultralytics YOLOv8 Docs](https://docs.ultralytics.com/yolov5/tutorials/running_on_jetson_nano/#install-pytorch-and-torchvision) trên github [ultralytics/yolov5: YOLOv5 🚀 in PyTorch > ONNX > CoreML > TFLite (github.com)](https://github.com/ultralytics/yolov5) bản 1.10.0 và torchvision 0.11.1

Opencv:4.1.1 cài theo vid [(1623) Install Opencv 4.1 on Nvidia Jetson Nano - YouTube](https://www.youtube.com/watch?v=SSzZZ68jy_I&t=517s)

Các thư viện khác cài theo lệnh **sudo python3 -m pip install** **(tên thư viện==phiên bản-nếu có)**

**B3: Luyện mô hình tạo file pt mới đuôi s**

**Cách 1:** Link hướng dẫn: https://www.youtube.com/watch?v=2TIhdai0f6M&t=674s

Thực hiện tạo file trên tool google colab:

Clone code model về:

!git clone https://github.com/ultralytics/yolov5.git

%cd yolov5/

Download data: !pip install roboflow

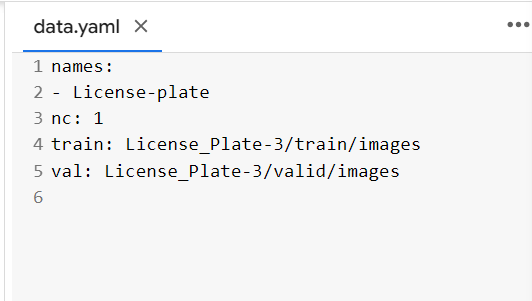
from roboflow import Roboflow

rf = Roboflow(api\_key="SHqXH9Hthjol3EotLlHd")

project = rf.workspace("ctarg").project("license\_plate-naqg1")

dataset = project.version(3).download("yolov5")

Cập nhập lại đường link tại file data.yaml bằng cách copy lại link image của folder train và val.



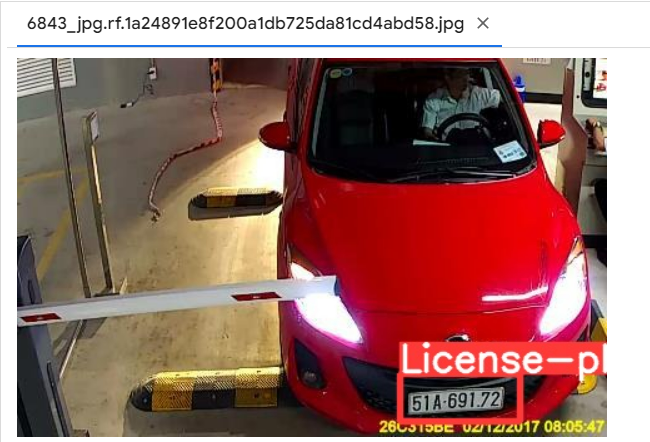
Train model bằng lệnh: (chú ý chỉnh về chạy trên GPU)

!python train.py --img 640 --batch 16 --epochs 100 --data /content/yolov5/License\_Plate-3/data.yaml --weights yolov5s.pt --nosave --cache

Cuối cùng chạy thử trên bộ dữ liệu test:

!python detect.py --weights /content/yolov5/runs/train/exp2/weights/last.pt --source /content/yolov5/License\_Plate-3/test/images

KẾt quả:



Link source: https://colab.research.google.com/drive/1HH4WNWfx14slBRhS0r9N0qOgsQv18PJl?usp=sharing

**Cách 2:** sử dụng mô hình yolo v5 đã wrap up thành tools, call lại thư viện và luyện lại mô hình.

from ultralytics import YOLO

model = YOLO("yolov5s.pt")

model.train(data="License\_Plate-3/data.yaml", epochs=3)

metrics = model.eval()

model.export(format="onnx")

Link: <https://github.com/ultralytics/yolov5>

Tuần 2 : 4/12-9/12

**Quy trình để OCR - nhận diện và đọc tốt biển số:**

**+B1 : Dùng model đã huấn luyện crop được vùng biển trên ảnh bằng tọa độ**

**+B2 : Tiền xử lý ảnh biển ( chuyển về ảnh xám , giảm nhiễu , làm mờ )**

**+B3 : Sử dụng bộ lọc Canny để tìm Contours**

**+B4 : Lọc ra contour có 4 tọa độ , tạo ảnh với tất cả các pixel đen trừ phần bên trong 4 tọa độ là các pixel trắng**

**+B5 : Thực hiện phép toán AND bit trên ảnh thu được từ bước 1 với ảnh tạo ra từ bước 4 thu được ảnh biển với nền ngoài đen**

**+B6 : Trích xuất tọa độ tại các pixel trắng ở góc ta thu được tọa độ của biển**

**+B7 : Tiến hành crop**

**Kết quả :**

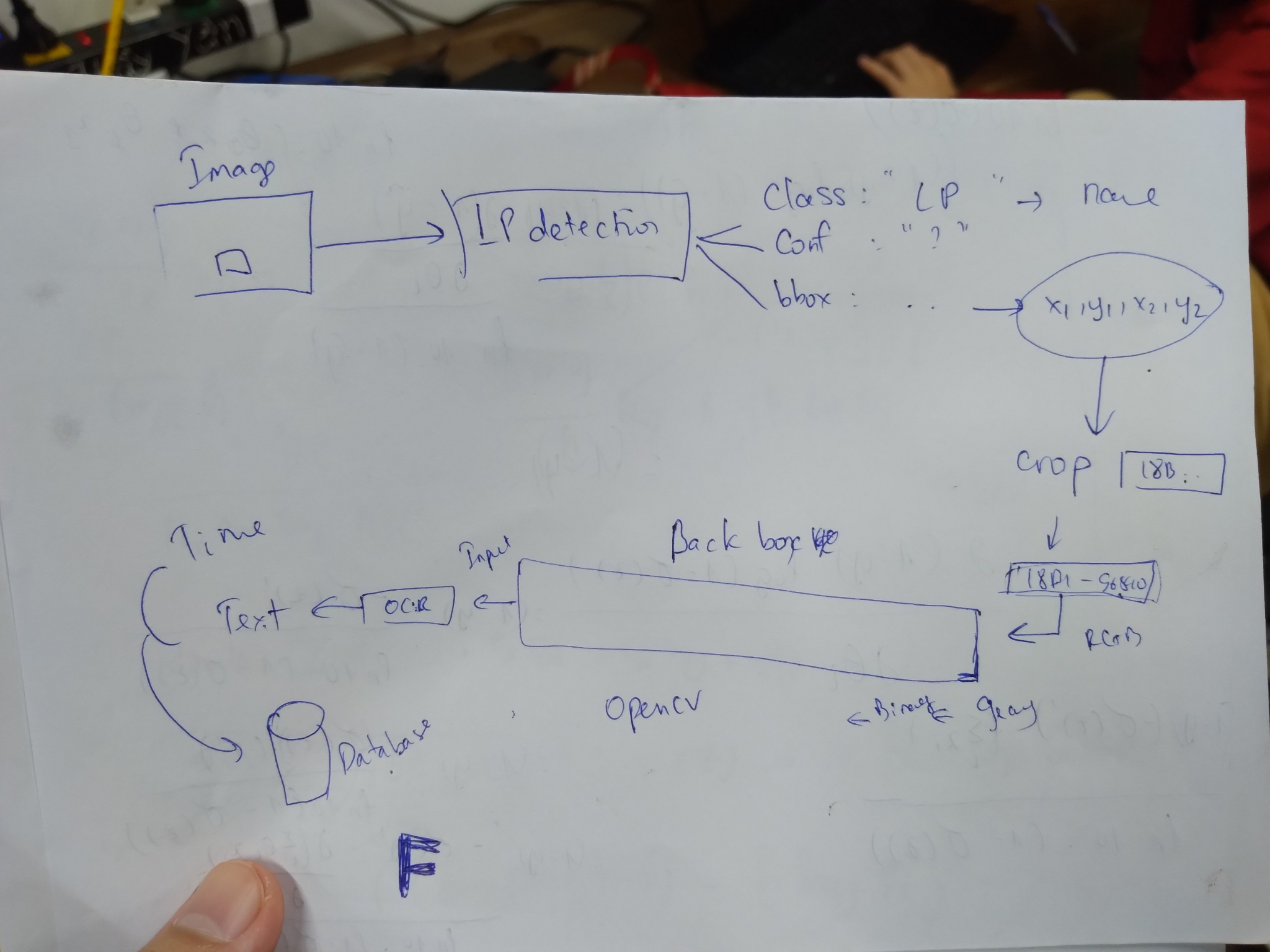
****

****

**+B8 : OCR**

**+Kết quả**

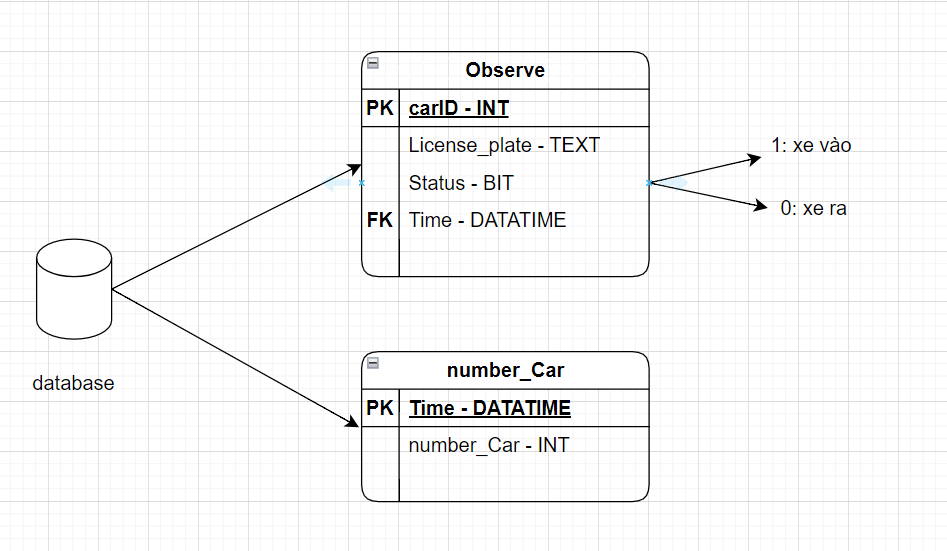
****



## 

**Thiết kế database mini cho camera**

Sử dụng database nhỏ phù hợp với camera Sqlite



**Trong đó:**

* carID: mỗi car đi vào xe được gắn 1 id riêng để phân biệt.
* License\_plate: biển số xe được detect.
* Status: trạng thái xe đi vào hay đi ra.
* Time: thời gian thực tế khi xe đi vào, đi ra.
* number\_Car: số lượng xe trong bãi gửi xe.

Hai bảng kết nối qua khóa phụ Real-time.

## Vướng mắc

Ko chuyển được từ file .pt sang file .engine:*./yolov5\_det -s best.wts best.engine s -> Đã xử lý xong*

→ Sửa thành công bằng cách sửa lại numclass = 1 - do bài toán của mình chỉ có 1 dối tượng là car (Khác với trong video hướng dẫn là 3).

Không chạy được file deploy.py trên jetson do lỗi đường dẫn

→ Sửa thành công bằng cách cài

## Lưu ý - kinh nghiệm xương máu

Nhớ sửa numclass = 1

Nhớ sửa numclass = 1

Nhớ sửa numclass = 1

Nhớ sửa numclass = 1

Nhớ sửa numclass = 1

## 

## 

## 

## Công việc tiếp theo

- Tiền xử lý ảnh biển đã detect

- Đọc biển bằng OCR

- Tạo database lưu thông tin biển

- Thiết kế giao diện (optional)

# 

# 

# Tổng quan