ĐỀ XUẤT MÔ HÌNH YOLO V5 ỨNG DỤNG TRONG NHẬN DIỆN BIỂN SỐ XE

***Nguyễn Thành Lợi*[*], *Đào Xuân Phúc*[†], *Nguyễn Thị Tố Uyên*[‡], *Nguyễn Hữu Phát*[§]**

Ngày tòa soạn nhận được bài báo: 05/12/2022
Ngày nhận kết quả phản biện đánh giá: 05/06/2023
Ngày bài báo được duyệt đăng: 28/06/2023

DOI: 10.59266/houjs.2023.277

Tóm tắt: *Deep learning-học sâu lấy ý tưởng từ bộ não sinh học, các mô hình học sâu xây dựng các thuật toán giúp máy duy nghĩ và xử lý thông tin giống như bộ não con người. Các mô hình, thuật toán học sâu phát triển ngày càng rộng rãi và được ứng dụng nhiều vào thực tiễn nhằm giảm thiểu tối đa sức lao động của con người. Bài báo trình bày về các vấn đề liên quan đến mô hình Yolov5, bao gồm nguyên lý hoạt động, áp dụng mô hình để đào tạo cho dữ liệu từ đó nhận diện biển số xe và đánh giá mô hình. Kết quả chỉ ra mô hình có độ chính xác cao chứng tỏ tính khả thi khi ứng dụng trong thực tế.*

Từ khóa: *Yolov5, Yolov2, Yolov3, Faster-RCNN, SVM, License-Plate-Recognition.*

I. Đặt vấn đề

Trong thời đại công nghệ thông tin hiện nay, việc sử dụng xe cộ trở nên ngày càng phổ biến. Để quản lý và kiểm soát giao thông, việc nhận diện biển số xe tự động đóng vai trò quan trọng. Nhận diện biển số xe thông qua công nghệ hình ảnh và xử lý ảnh đã thu hút sự quan tâm của nhiều nhà nghiên cứu và các tổ chức liên quan. Mặc dù đã có sự tiến bộ đáng kể trong công nghệ nhận diện hình ảnh, nhưng nhận diện biển số xe vẫn là một thách thức đối với các hệ thống giám sát giao thông. Có nhiều yếu tố gây khó khăn trong quá trình nhận diện, chẳng hạn như thay đổi về ánh sáng, góc nhìn, độ mờ, kích thước và font chữ biển số xe. Việc xử lý và phân tích dữ liệu từ các hình ảnh chứa biển số xe đòi hỏi các thuật toán phức tạp và cần được tối ưu hóa để đạt được độ chính xác cao và đáng tin cậy.

Đề tài này tập trung vào nghiên cứu và phát triển hệ thống nhận diện biển số xe tự động, có khả năng ứng dụng rộng rãi trong các lĩnh vực như quản lý giao thông, an ninh, định danh xe cộ, và giám sát đô thị.

[*] Trường Điện-Điện tử, Đại học Bách Khoa Hà Nội
[†] Khoa Điện - Điện tử, Trường Đại học Mở Hà Nội
[‡] Khoa Điện - Điện tử, Trường Đại học Mở Hà Nội
[§] Trường Điện-Điện tử, Đại học Bách Khoa Hà Nội

II. Các nghiên cứu liên quan

Với kỹ thuật xử lý ảnh sử dụng các model để nhận diện biển số xe đã có nhiều công bố, bài báo được đưa ra. Bảng dưới đây là một số nghiên cứu với các model khác nhau.

License Plate Detection				
	Precision	**Recall**	**F1-score**	**mAP@0.5**
Tiny YOLOv4	96.6±1.0	97.6±1.0	96.8±0.7	97.0±0.5
Tiny YOLOv3	93.0±1.4	92.0±1.4	92.6±1.0	95.0±0.7
Faster-RCNN	96.2±1.3	96.4±1.0	96.2±0.7	96.0±0.7
YOLOv2	93.4±1.3	93.0±1.7	93.4±1.0	95.3±0.8
YOLOv3	97.6±0.5	97.6±0.5	97.6±0.5	98.0±0.4
YOLOv4	97.4±1.0	97.4±1.2	97.4±0.5	98.5±0.3
Character Recognition				
	Precision	**Recall**	**F1-score**	**mAP@0.5**
Tiny YOLOv4	94.0±2.0	94.4±2.2	94.4±1.5	93.8±0.5
Tiny YOLOv3	89.8±1.2	92.0±1.7	91.4±1.0	92.4±1.0
Faster-RCNN	93.0±1.4	93.2±1.6	93.2±1.5	93.0±1.0
YOLOv2	91.6±1.3	91.8±1.7	91.8±1.1	92.8±0.7
YOLOv3	97.6±0.8	95.4±1.6	96.8±1.0	97.0±0.4
YOLOv4	97.2±0.7	97.0±1.1	97.0±0.6	98.3±0.5

Bảng 1: Một số đánh giá kết quả từ các model YOLO [1]

	OpenALPR EU	**OpenALPR BR**	**SSIG Test**	**AOLP RP**	**Proposed CD-HARD**	**Average**
Ours	93.52%	91.23%	**88.56%**	**98.36%**	**75.00%**	**89.33%**
Ours (no artf.)	92.59%	88.60%	84.58%	93.29%	73.08%	86.43%
Ours (unrect.)	94.44%	90.35%	87.81%	84.61%	57.69%	82.98%
Commercial systems						
OpenALPR	**96.30%**	85.96%	87.44%	69.72%*	67.31%	81.35%
Sighthound	83.33%	**94.73%**	81.46%	83.47%	45.19%	77.64%
Amazon Rekog.	69.44%	83.33%	31.21%	68.25%	30.77%	56.60%
Literature						
Laroca et al. [17]	-	-	85.45%	-	-	-
Li et al. [18]	-	-	-	88.38%	-	-
Li et al. [19]	-	-	-	83.63%	-	-
Hsu et al. [10]	-	-	-	85.70%**	-	-

Bảng 2: Đánh giá so sánh về độ chính xác của model Wpod-net và các model khác [2]

III. Phương pháp đề xuất

3.1. Mô hình bài toán

Bài toán nhận diện biển số xe có mô hình chung như sau:

• Khối thứ nhất: là khối phát hiện biển số xe, ta sẽ thu thập dữ liệu thông qua video, camera, hoặc các ảnh về các phương tiện giao thông có chứa biển số xe từ đó huấn luyện mô hình để có thể phát hiện và khoanh vùng tạo bounding box cho vị trí biển số trong những bức ảnh hoặc video đó, các bức ảnh thường đa dạng về mặt dữ liệu có thể chứa nhiều biển số xe cũng như góc chiếu, tỉ lệ kích thước, độ nét và góc khuất là khác nhau.

• Khối thứ hai: là khối phân tách kí tự trong biển số từ đó nhận dạng từng kí tự, cuối cùng trả ra kết quả là khu vực khoanh vùng biển số và kí tự trên biển số đó

Bài toán đã kết hợp hai khối trên, khối thứ nhất khoanh vùng được khu vực chứa biển số xe, từ đầu ra khối thứ nhất trở thành đầu vào của khối từ hai. khối thứ hai sẽ thực hiện nhận dạng từng kí tự trong khu vực khoanh vùng đó để cho ra output của bài toán.

Hình 1 là sơ đồ khối tổng quát của bài toán nhận diện biển số xe

Hình 1: Sơ đồ khối hệ thống

3.2. Yêu cầu dữ liệu

• Hình ảnh các phương tiện tham gia giao thông, các loại xe trên đường phố.

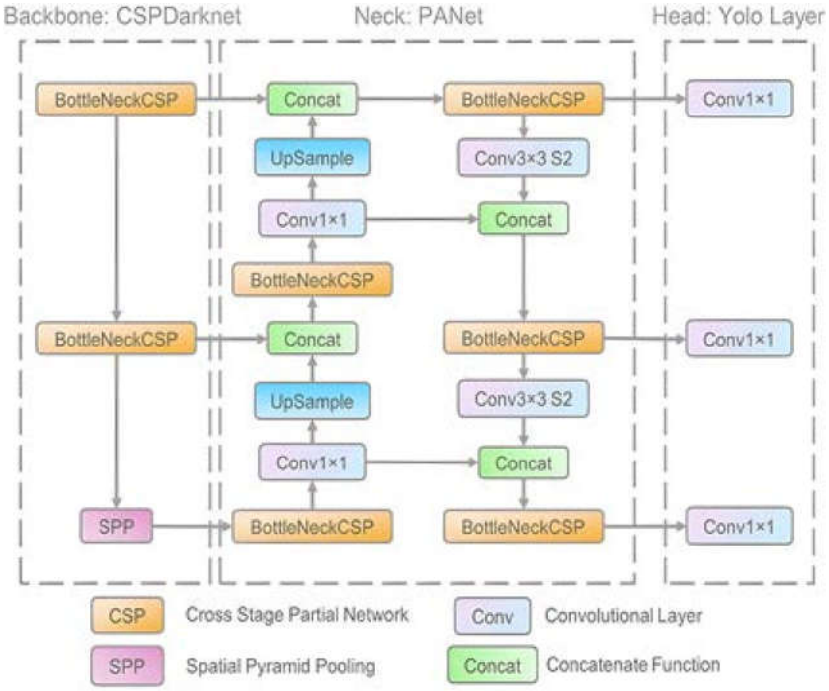
• Hình ảnh các loại biển số xe chứa kí tự rõ nét.

• Hình ảnh các phương tiện mà có phần đầu và phần đuôi xe chứa biển số và rõ nét kí tự trên biển

Từ những yêu cầu trên chúng tôi đã đi thu thập được lượng data phong phú đa dạng giúp việc đào tạo dữ liệu trở nên hiệu quả và chính xác.

3.3. Kiến trúc mô hình yolo V5

YOLOv5 là phiên bản thứ 5 của YOLO [3]. Mô hình này được phát triển bởi công ty Ultralytics và được sử dụng để phát hiện đối tượng trong hình ảnh và video. Tuy nhiên, phiên bản này không có bài báo chính thức và chỉ có repository trên Github. YOLOv5 đã cho thấy đây là phiên bản có thể cải thiện độ chính xác, tốc độ và hiệu suất so với YOLOv4 [4]. Thêm vào đó YOLOv5 còn giảm thiểu kích thước của mô hình để có thể chạy trên các thiết bị tầm trung và điện thoại thông minh.



Hình 2: Kiến trúc của Yolov5 [5]

Kiến trúc: Kiến trúc của YOLOv5 gồm ba phần chính:

• **Backbone:** Dựa trên kiến trúc CSPNet (Cross-Stage Partial networks) [6]. Kiến trúc CSP được dùng để xây dựng mô hình lưới mới, giúp cải thiện tốc độ và độ chính xác của mô hình so với các phiên bản tiền nhiệm. Kiến trúc CSP chia mạng CNN thành các tầng (stage) riêng biệt, với mỗi tângf có một phần tiền xử lý (pre- processing) và một phần hậu xử lý (postprocessing). Việc tách CNN thành các tầng nhỏ hơn giúp làm giảm độ phức tạp tính toán và giúp huấn luyện mô hình nhanh hơn.

• **Neck:** Sử dụng một mô hình trích xuất đặc trưng hiệu quả để giảm độ phức tạp và tăng tốc độ xử lý. Mô hình này được xây dựng trên CSPNet với các lớp phân tíchvà tích hợp đặc trưng, giúp trích xuất các đặc trưng của ảnh một cách chính xác và hiệu quả.

• Head: Được xây dựng trên kiến trúc của YOLOv3 nhưng được cải tiến và tối ưu tốt hơn để đạt được độ chính xác và tốc độ xử lý cao hơn.

3.4. Model Training yolo V5

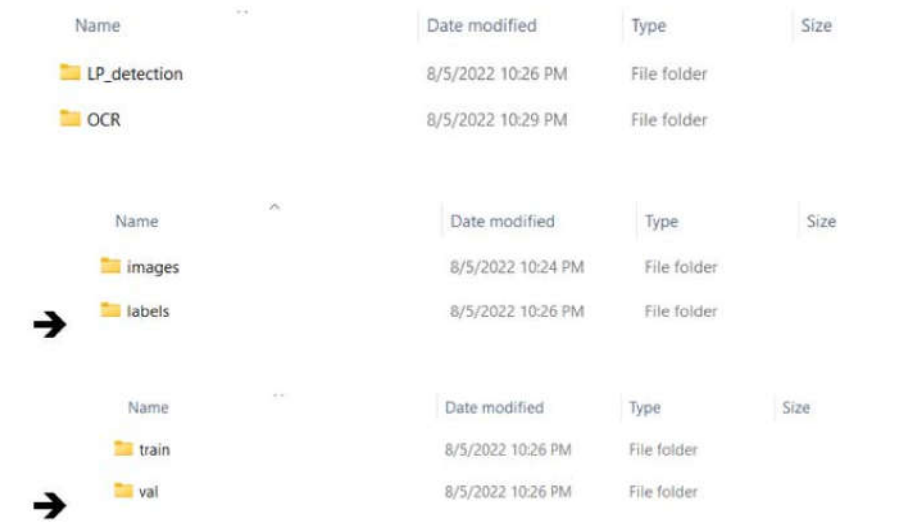
Model traning sử dụng để nhận diện vị trí của biển số trong ảnh và đồng thời dùng cả model để nhận diện chữ cái của biển số. Để phát hiện đối tượng nào đó, bước đầu tiên chúng ta cần thu thập thật nhiều ảnh của chúng với nhiều tình huống trường hợp giúp cho data trở nên phong phú đa dạng rồi sau đó chia thành các tập train/val. Tool dán nhãn phổ biến được sử dụng là labellmg:

Dataset khi xây dựng sẽ có hai mục lớn là Images và Labels. Trong mỗi thư mục lớn trên sẽ chứa 2 thư mục con bao gồm thư mục train và val. khi đó dữ liệu sẽ được chia với quy tắc tỷ lệ dữ liệu trong thư mục train chiếm khoảng 80% và tỷ lệ dữ liệu trong thư mục val chiếm khoảng 20%. Thư mục Images sẽ bao gồm ảnh của dữ liệu, thư mục Labels sẽ chứa các nhãn được đánh của các ảnh dữ liệu trên.

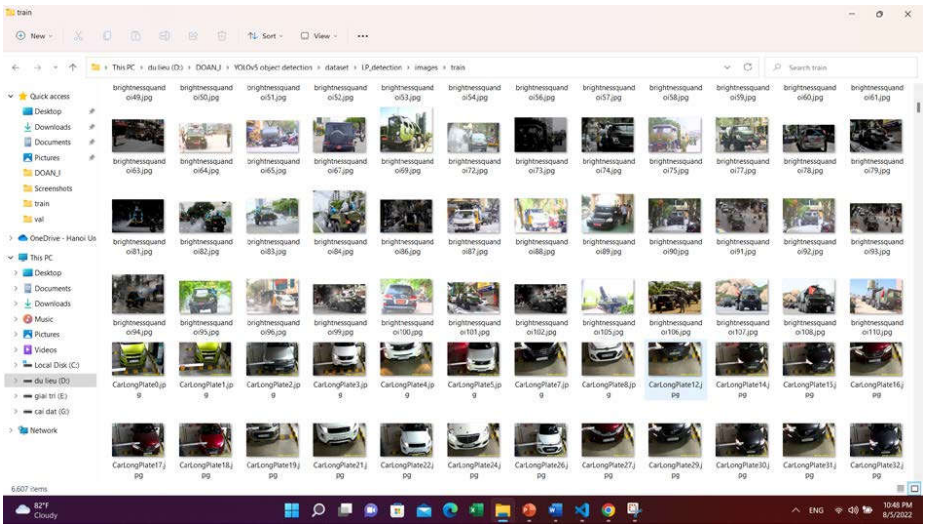
IV. Thực nghiệm

4.1. Datasets

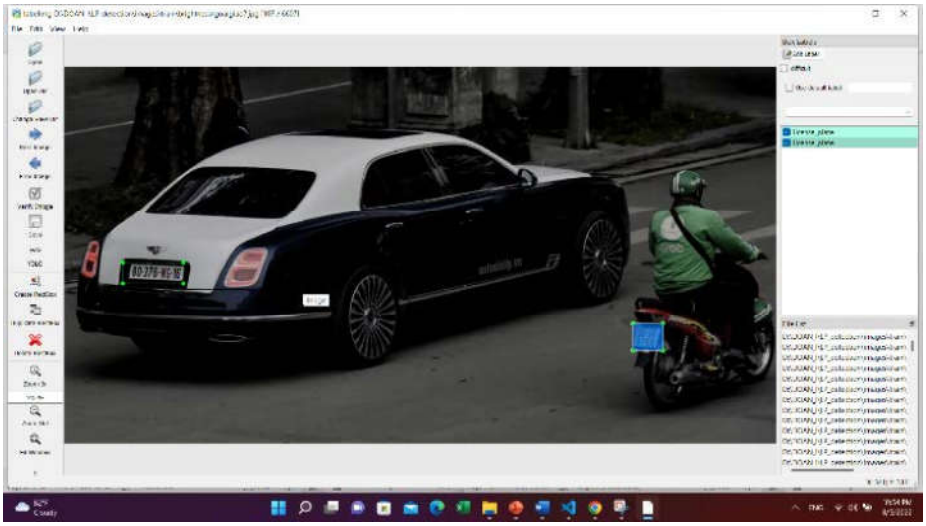
Trong mô hình nhận diện biển số xe này chúng tôi sẽ xây dựng 2 bộ dataset đó là bộ dữ liệu về biển số xe và bộ dữ liệu về ký tự trên biển số. Với bộ dữ liệu về biển số xe (LP_detection) ở thư mục Images của bộ dữ liệu này sẽ bao gồm ảnh của các loại xe có chứa biển số xe. Còn thư mục Labels sẽ chứa nhãn khoang vùng biển số trong bức hình đó và được lưu dưới dạng file: txt. Bộ dữ liệu thứ hai chính là dữ liệu tập hợp toàn bộ ký tự của biển số xe (OCR). Tương tự như bộ dataset trên ở thư mục Images sẽ bao gồm tập hợp ảnh các biển số sau khi đã được cắt ra, trong thư mục Labels sẽ chứa các nhãn khoanh vùng của từng ký trên từng biển số. Dưới đây là một số hình ảnh về quá trình xây dựng bộ dữ liệu của chúng tôi. Ví dụ dữ liệu như trên hình 3 đến 8.



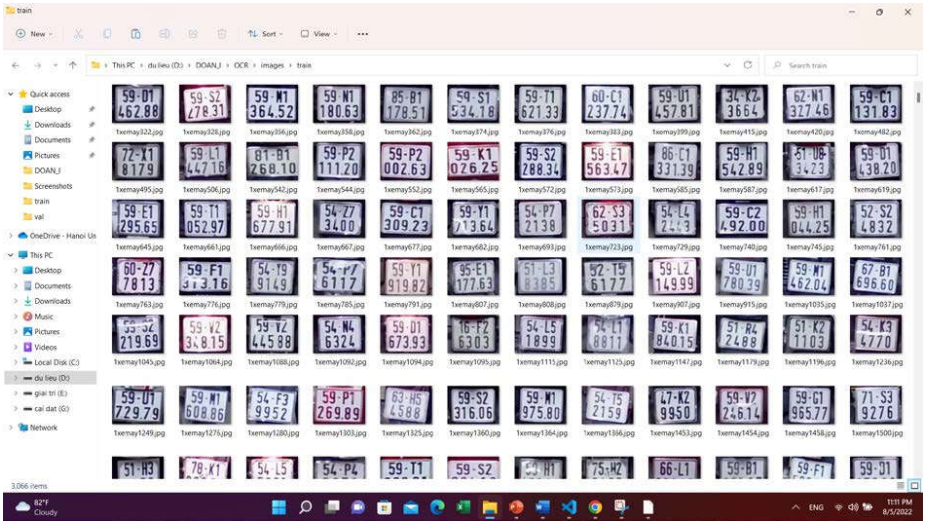
Hình 3: Cấu trúc phân bố dữ liệu



Hình 4: Kho dữ liệu phương tiện chứa biển số

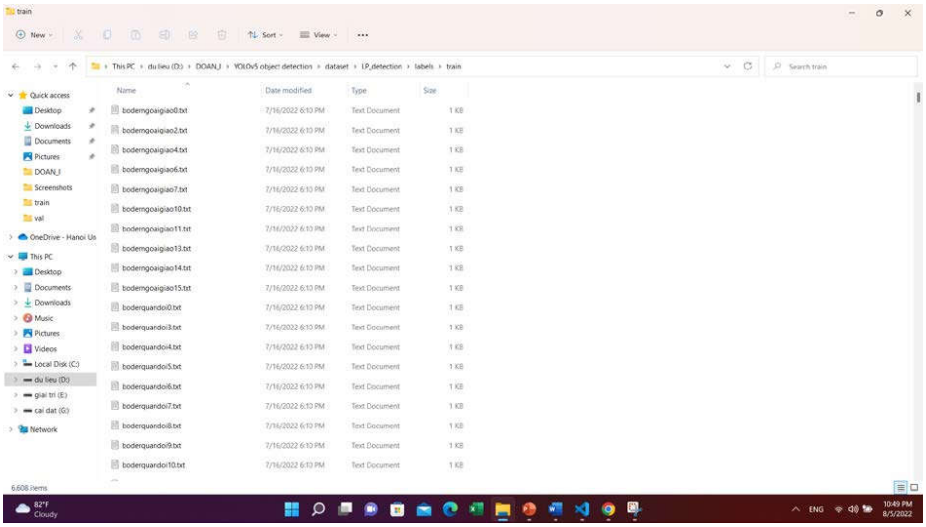


Hình 5: Quá trình đánh nhãn biển số



Hình 6: Kho dữ liệu biển số sau khi tách ra từ phương tiện

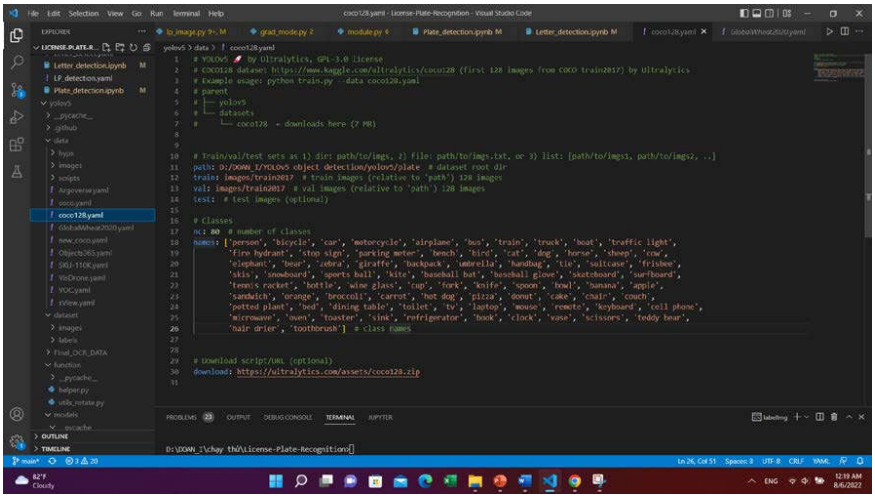
Hình 7: Quá trình đánh nhãn kí tự biển trong biển số



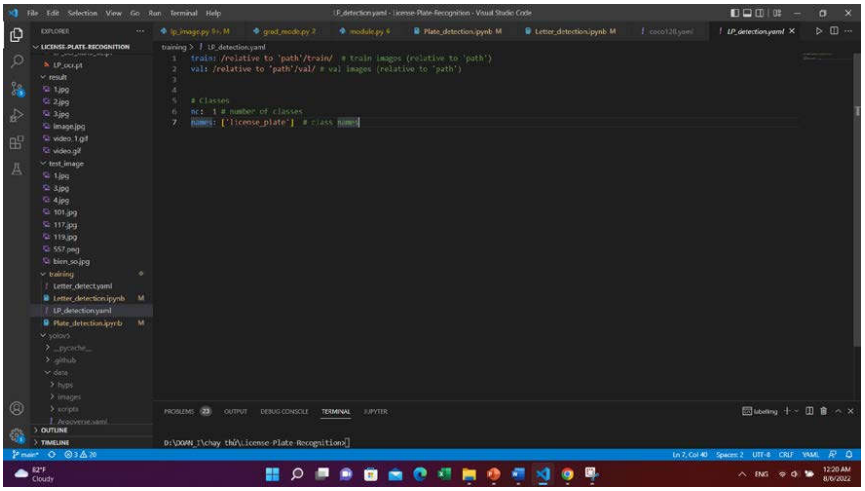
Hình 8: File dữ liệu txt trong khi đánh nhãn

4.2. Train

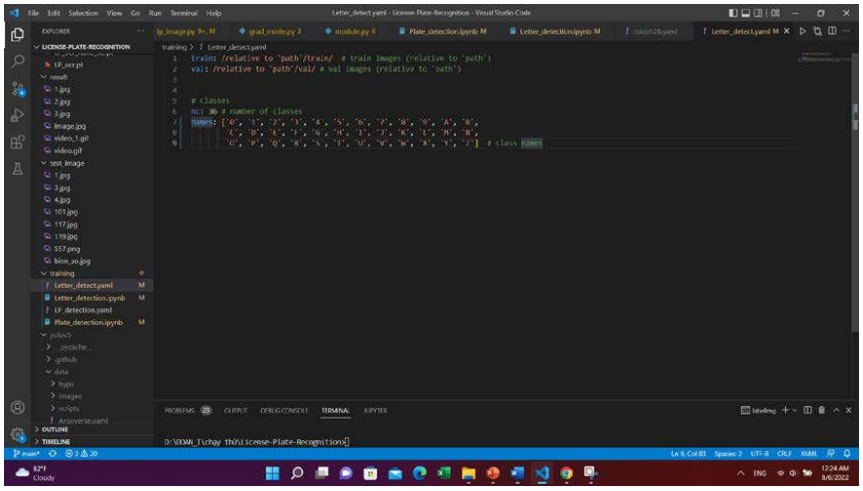
Thay đổi tham số đầu vào và truyền đối số với dữ liệu của mình như trên hình 9, 10, và 11.



Hình 9: Tham số gốc của model



Hình 10: Thay đổi tham số với đầu vào là nhãn biển số xe (License_plate)

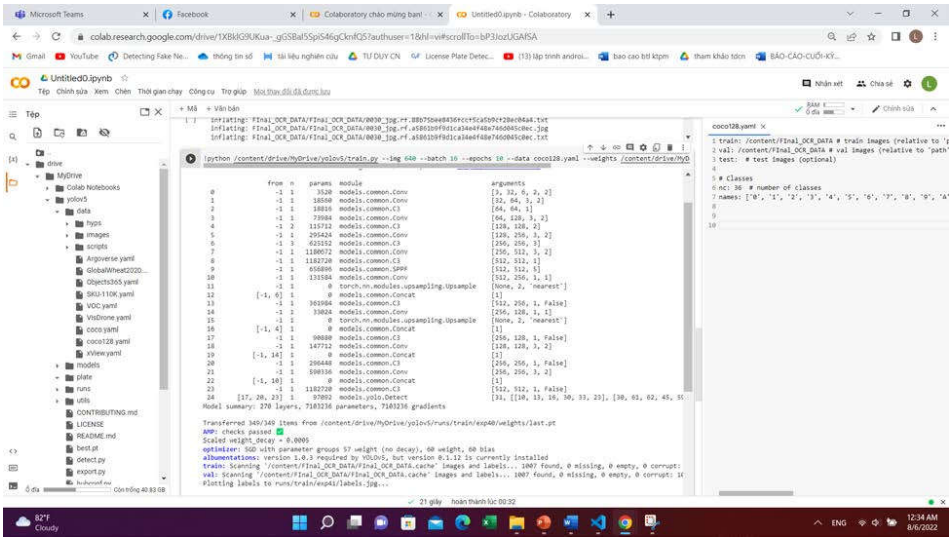


Hình 11: Thay đổi tham số với đầu vào chứa nhãn các ký tự của biển số (OCR)

Google Colab và Hugging Face là hai web rất hữu ích khi học tập AI. Colab cho phép bạn viết code python, chạy trên web với GPU miễn phí (giới hạn thời gian) và chia sẻ code dễ dàng.

Hugging Face là web chứa rất nhiều mô hình, dữ liệu train miễn phí và cũng dễ dàng để sử dụng.

Trong bài báo này chúng tôi sử dụng Colab để train mô hình như trên hình 12.



Hình 12. Một đoạn code khi train mô hình được chạy trên Colab

4.3. Các thông số đánh giá

Trong quá trình xây dựng một mô hình Machine Learning, một phần không thể thiếu để xét xem mô hình có chất lượng tốt hay không chính là đánh giá mô hình. Đánh giá mô hình giúp chúng ta chọn lựa được các mô hình phù hợp với bài toán cụ thể. Để có thể áp dụng đúng thước đo đánh giá mô hình phù hợp, chúng ta cần hiểu bản chất, ý nghĩa cũng như các trường hợp sử dụng nó

• Accuracy là tỉ lệ giữa số điểm được phân loại đúng và tổng số điểm. Accuracy chỉ phù hợp với các bài toán mà kích thước các lớp dữ liệu là tương đối như nhau.

• Confusion matrix giúp có cái nhìn rõ hơn về việc các điểm dữ liệu được phân loại đúng/sai như thế nào.

• True Positive (TP): số lượng điểm của lớp *positive* được phân loại đúng là *positive*.

• True Negative (TN): số lượng điểm của lớp *negative* được phân loại đúng là *negative*.

• False Positive (FP): số lượng điểm của lớp *negative* bị phân loại nhầm thành *positive*.

• False Negative (FN): số lượng điểm của lớp *positiv* bị phân loại nhầm thành *negative*

• True positive rate (TPR), false negative rate (FNR), false positive rate (FPR), true negative rate (TNR) như trên hình 14.

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

• True Positive (TP): Là tập hợp các điểm mà đối tượng thực tế thuộc vào lớp positive và mô hình dự đoán ra đúng nó thuộc vào lớp positive.

• True Negative (TN): Là tập hợp các điểm mà đối tượng thực tế thuộc vào lớp negative và mô hình dự đoán ra đúng đối tượng thuộc vào lớp negative.

• False Positive (FP): Là tập hợp các điểm mà đối tượng thực tế thuộc vào lớp negative nhưng mô hình dự đoán ra đối tượng thuộc vào lớp positive.

• False Negative (FN): Là tập hợp các điểm mà đối tượng thực tế thuộc vào lớp positive nhưng mô hình dự đoán ra đối tượng thuộc vào lớp negative.

• Để tính toán mAP, đầu tiên ta phải tính toán giá trị Precision và Recall cho từng lớp đối tượng, sau đó tính toán diện tích dưới đường cong Precision-Recall cho mỗi lớp.

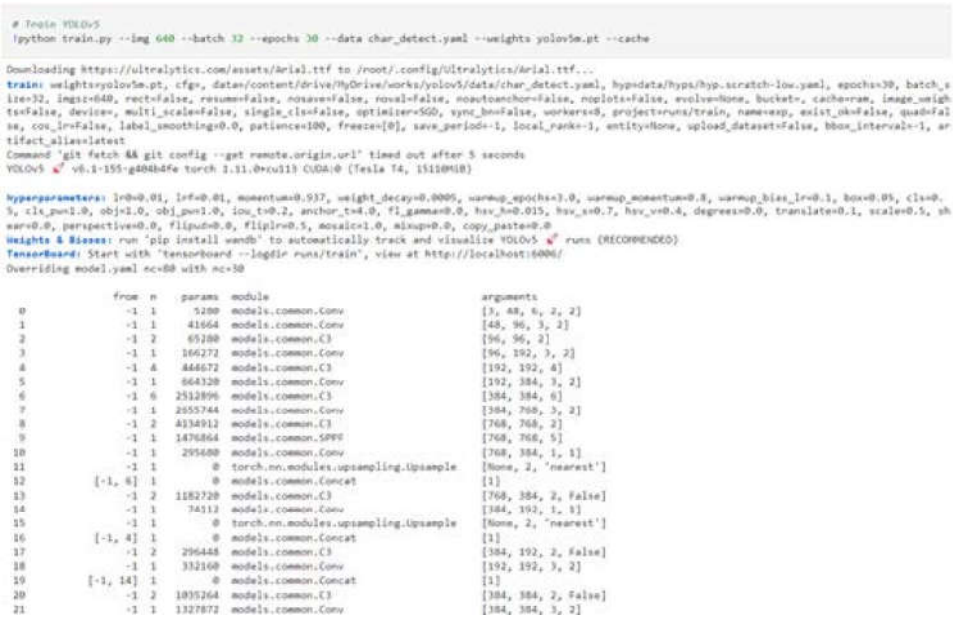
$$AP = \sum_{n} (R_n - R_{n-1}).P_n \qquad (3)$$

Trong đó: P_n và R_n lần lượt là giá trị Precision và Recall tại điểm có thứ tự n trên PR Curve. Điểm đầu tiên trên PR Curve là (0, 1) và điểm cuối cùng là (1, 0).

Cuối cùng, ta tính toán trung bình của các diện tích này trên toàn bộ các lớp đối tượng để có được giá trị mAP.

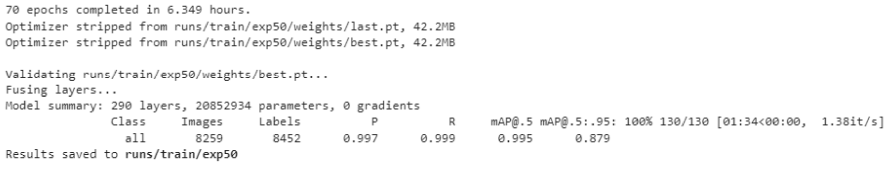
$$mAP = \frac{1}{n} \sum_{i=1} AP_i \qquad (4)$$

Một mô hình object detection tốt sẽ có một giá trị mAP cao, tức là mô hình có thể phát hiện các lớp đối tượng đúng và không gây ra quá nhiều sai sót.



Hình 13. Quá trình thực hiện train mô hình được chạy trên Colab

4.4. Kết quả



Hình 14. Kết quả train phát hiện biển số xe

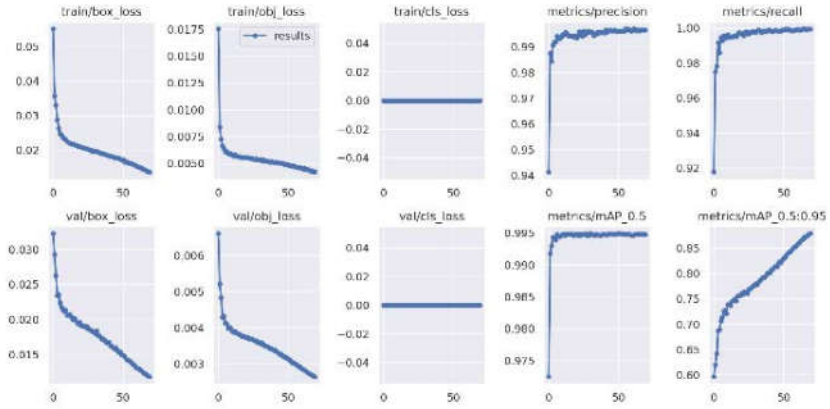
Kết quả train tập dữ liệu bao gồm train và val với tổng cộng 8259 ảnh và 8452 biển số xe được đánh nhãn sau 70 epochs như trên hình 15 đạt được các thông số như sau:

Precision đạt: 99.7%

Recall đạt: 99.9%

mAP@.5 đạt: 99.5%

mAP@.5: .95 đạt: 87.9%



Hình 15: Biểu đồ train 70 epochs

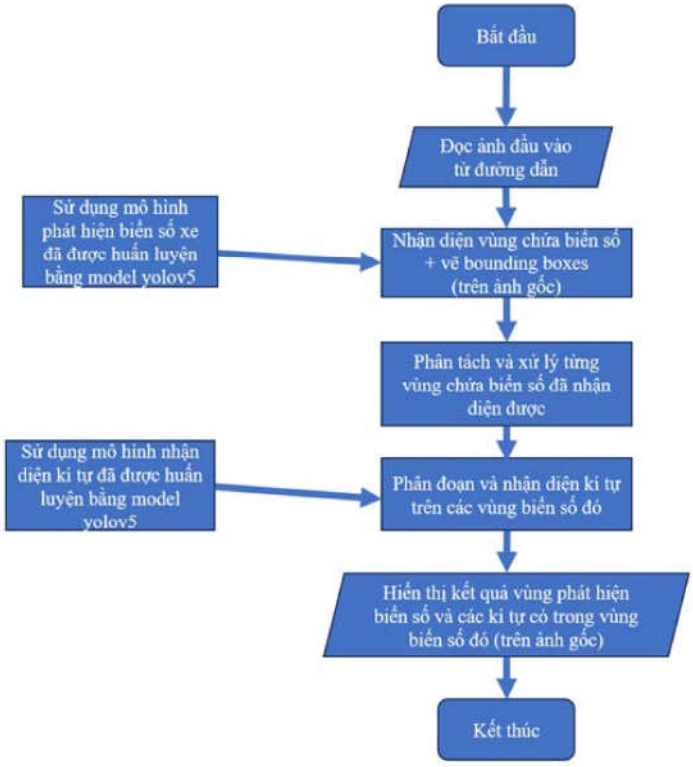
Từ những thông số trên ta có thể thấy mô hình train ứng với tập dữ liệu cho kết quả khá tốt như trên hình 15.

```
     Epoch   gpu_mem       box       obj       cls    labels  img_size
     28/29     12.1G   0.01948   0.04702   0.00504       412       640: 100% 96/96 [01:17<00:00,  1.23it/s]
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:06<00:00,  1.92it/s]
                 all        767       6236      0.979      0.969      0.983       0.76

     Epoch   gpu_mem       box       obj       cls    labels  img_size
     29/29     12.1G   0.01922    0.0466  0.004922       408       640: 100% 96/96 [01:17<00:00,  1.23it/s]
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:06<00:00,  1.95it/s]
                 all        767       6236      0.976      0.974      0.983      0.763

30 epochs completed in 0.725 hours.
Optimizer stripped from runs/train/exp64/weights/last.pt, 42.5MB
Optimizer stripped from runs/train/exp64/weights/best.pt, 42.5MB

Validating runs/train/exp64/weights/best.pt...
Fusing layers...
Model summary: 290 layers, 20070123 parameters, 0 gradients
               Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:09<00:00,  1.29it/s]
                 all        767       6236      0.976      0.974      0.983      0.763
                   1        767        836      0.991      0.972      0.984      0.674
                   2        767        541      0.996      0.991      0.994      0.798
                   3        767        423      0.998       0.98      0.995      0.802
                   4        767        425      0.993      0.993      0.992      0.752
                   5        767        754      0.985      0.995      0.991      0.797
                   6        767        428      0.993      0.986      0.994      0.796
                   7        767        418      0.986      0.984       0.99      0.777
                   8        767        373      0.984      0.981      0.992       0.78
                   9        767        630      0.994      0.991      0.994      0.799
                   A        767         64      0.955          1      0.993      0.723
                   B        767        153      0.967      0.963      0.986      0.787
                   C        767         52      0.994      0.923      0.965      0.757
                   D        767         56      0.914      0.929      0.936      0.731
                   E        767         40      0.973          1      0.995      0.808
                   F        767         72      0.984          1      0.995      0.776
                   G        767         68          1      0.976      0.995       0.71
                   H        767         39      0.986      0.872      0.979      0.786
                   K        767         34      0.989          1      0.995      0.738
                   L        767         42          1      0.981      0.995      0.738
                   M        767         60      0.932      0.907      0.972      0.776
                   N        767         22      0.956      0.999      0.975      0.754
                   P        767         34      0.995          1      0.995      0.751
                   S        767         40      0.972          1      0.995       0.75
                   T        767         33      0.985       0.97      0.994      0.783
                   U        767         31      0.992          1      0.995      0.788
                   V        767         17      0.923      0.941      0.964      0.753
                   X        767         26      0.996          1      0.995       0.74
                   Y        767          9      0.873      0.889      0.872      0.728
                   Z        767         55      0.998          1      0.995      0.756
                   0        767        461      0.992      0.991       0.99      0.793
Results saved to runs/train/exp64
```

Hình 16: Kết quả train nhận dạng kí tự

Kết quả train nhận diện kí tự của tập dữ liệu val với 767 ảnh và số lượng nhãn được đánh cho từng kí tự đều cho ra các kết quả dự đoán precision và recall với độ chính xác là rất cao như trên hình 16.

Sơ đồ khối thuật toán:



Hình 18. Sơ đồ khối thuật toán

```
from PIL import Image
import cv2
import torch
import math
import function.utils_rotate as utils_rotate
from IPython.display import display
import os
import time
import argparse
import function.helper as helper

ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=True, help="D:\DOAN_1\YOLOV5 object detection\yolov5\License-Plate-Recognition\test_image\bien_so.jpg")
args = ap.parse_args()

yolo_LP_detect = torch.hub.load('yolov5', 'custom', path='model/LP_detector.pt', force_reload=True, source='local')
yolo_license_plate = torch.hub.load('yolov5', 'custom', path='model/LP_ocr.pt', force_reload=True, source='local')
yolo_license_plate.conf = 0.60

img = cv2.imread(args.image)
plates = yolo_LP_detect(img, size=640)

plates = yolo_LP_detect(img, size=640)
list_plates = plates.pandas().xyxy[0].values.tolist()
list_read_plates = set()
if len(list_plates) == 0:
    lp = helper.read_plate(yolo_license_plate,img)
    if lp != "unknown":
        cv2.putText(img, lp, (7, 70), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (36,255,12), 2)
        list_read_plates.add(lp)
else:
    for plate in list_plates:
        flag = 0
        x = int(plate[0]) # xmin
        y = int(plate[1]) # ymin
        w = int(plate[2]) - plate[0]) # xmax - xmin
        h = int(plate[3]) - plate[1]) # ymax - ymin
        crop_img = img[y:y+h, x:x+w]
        cv2.rectangle(img, (int(plate[0]),int(plate[1])), (int(plate[2]),int(plate[3])), color = (0,0,225), thickness = 2)
        cv2.imwrite("crop.jpg", crop_img)
        rc_image = cv2.imread("crop.jpg")
        lp = ""
        for cc in range(0,2):
            for ct in range(0,2):
                lp = helper.read_plate(yolo_license_plate, utils_rotate.deskew(crop_img, cc, ct))
                if lp != "unknown":
                    list_read_plates.add(lp)
                    cv2.putText(img, lp, (int(plate[0]), int(plate[1]-10)), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (36,255,12), 2)
                    flag = 1
                    break
            if flag == 1:
                break
cv2.imshow("frame", img)
cv2.waitKey()
cv2.destroyAllWindows()
```

Hình 19: Xây dựng đoạn code đưa ra output

Đoạn mã trên thực hiện liên kết hai khối: Đưa output của khối phát hiện biển số xe trở thành input của khối nhận diện kí tự. Cuối cùng output của hai khối sau khi liên kết đưa ra là ảnh biển số xe đã được nhận dạng với kí tự của biển số xe hiển thị dưới dạng text.

Mô tả chi tiết thuật toán trong đoạn mã trên:

1. Đọc và hiển thị ảnh gốc:

a. Đọc ảnh đầu vào từ đường dẫn đã được chỉ định.

b. Hiển thị ảnh gốc.

2. Dùng mô hình YOLOv5 để nhận diện vùng chứa biển số (License Plate):

a. Sử dụng mô hình đã được huấn luyện trước (LP_detector.pt) để nhận diện vùng chứa biển số.

b. Vẽ các hộp giới hạn (bounding boxes) cho vùng chứa biển số trên ảnh gốc.

3. Xử lý từng vùng chứa biển số nhận diện được:

a. Lặp qua từng vùng chứa biển số.

b. Cắt vùng chứa biển số từ ảnh gốc để có hình ảnh riêng biệt.

c. Tiến hành xử lý ảnh để chuẩn bị cho bước nhận dạng kí tự.

4. Sử dụng mô hình YOLOv5 nhận diện các kí tự trong vùng chứa biển số:

a. Sử dụng mô hình đã được huấn luyện trước (LP_ocr.pt) để nhận diện các kí tự trong vùng chứa biển số.

b. Tiến hành đối sánh và xử lý vùng ảnh cắt với các kí tự được nhận diện.

5. Hiển thị kết quả nhận diện biển số và kí tự lên ảnh gốc:

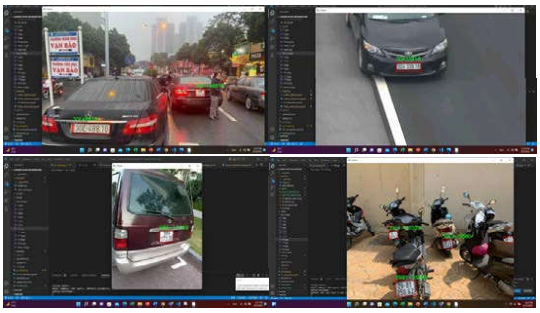
a. Nếu không tìm thấy vùng chứa biển số nào trên ảnh gốc, thì sử dụng mô hình YOLOv5 để nhận diện trực tiếp biển số và ghi kết quả lên ảnh gốc.

b. Nếu tìm thấy vùng chứa biển số, thì sử dụng mô hình YOLOv5 để nhận diện các kí tự trong từng vùng chứa và ghi kết quả lên ảnh gốc.

6. Hiển thị ảnh đã được xử lý:

a. Hiển thị ảnh gốc với kết quả nhận diện biển số và kí tự đã được ghi lên.

Khi kiểm tra mô hình với dữ liệu có sẵn là các ảnh được lưu dạng tệp có đuôi jpg, png chứa biển số xe thì kết quả thu được khá tốt như hình 19.



Hình 19. Kết quả kiểm tra nhận diện biển

V. KẾT LUẬN

Trong bài bài báo này tôi đã tập trung vào việc sử dụng model Yolov5 để ứng dụng vào bài toán nhận diện biển số xe, các vấn đề liên quan như mô hình xây dựng, quá trình xây dựng đào tạo dữ liệu, liên kết các khối để đưa ra output bài toán. Với dữ liệu đầu vào là hơn 8000 ảnh thì kết quả thu được của mô hình là chấp nhận được. Tuy nhiên mô hình của chúng tôi vẫn còn sai sót. Do đó hướng phát triển trong tương lai sẽ:

• Cần cải thiện dữ liệu train để kết quả được tăng độ chính xác hơn.

• Ứng dụng thêm các thuật toán khác để tăng độ chính xác của kết quả thu được.

• Từ đề tài ta có thể ứng dụng trực tiếp nó vào nhận diện biển số ở các bãi gửi xe, các trạm thu phí, các trạm gác cổng.

• Phát triển thêm các ứng dụng của đề tài trong lĩnh vực trí tuệ nhân tạo như robot đọc chữ. Hoặc phát triển thành máy phát hiện kí tự hỗ trợ những người khiếm thính, mù lòa….

Tài liệu tham khảo:

[1]. Muhammad Usama, Hafeez Anwar, Abbas Anwar, Saeed Anwar. Vehicle and License Plate Recognitionwith Novel Dataset for Toll Collection. arXiv:2202.05631v2 [eess.IV] 15 Nov 2022

[2]. S´ergio Montazzolli Silva and Cláudio Rosito Jung. License Plate Detection and Recognition in Unconstrained Scenarios. Doi: 10.1007/978-3-030-01258-8_36

[3]. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. arXiv:1506.02640v5 [cs.CV] 9 May 2016, DOI: 10.1109/CVPR.2016.91

[4]. Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934v1 [cs.CV] 23 Apr 2020

[5]. R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, "A forest fire detection system based on ensemble learning," Forests, vol. 12, no. 2, 2021. [Online]. Available: https://www.mdpi.com/1999-4907/12/2/217

[6]. Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu1, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. DOI: 10.1109/CVPRW50498.2020.00203

[7]. https://github.com/Marsmallotr/License-Plate-Recognition

[8]. https://github.com/ultralytics/yolov5

[9]. https://machinelearningcoban.com/2017/08/31/evaluation/

[10]. https://1upnote.me/post/2018/11/ds-ml-svm-mnist/

[11]. https://phamdinhkhanh.github.io/2020/03/09/DarknetAlgorithm.html

APPLYING YOLO V5 MODEL FOR DETECTING LICENSE PLATE

Nguyen Thanh Loi[¶], Dao Xuan Phuc[**],
Nguyen Thi To Uyen[††], Nguyen Huu Phat[‡‡]
Email: loi.nt192982@sis.hust.edu.vn

Abstract: *Deep learning is inspired by biological brains, deep learning models build algorithms that help machines think and process information like a human brain. Deep learning models and algorithms have been developed more and more widely and are widely applied to minimize human labour. The article presents the problems related to the Yolov5 model, including the operating principle, using the model to train data from which to recognize license plates and evaluate the model. The results show that the model has high accuracy, proving its feasibility when applied in practice.*

Keywords: *Yolov5, Yolov2, Yolov3, Faster-RCNN, SVM, License-Plate-Recognition*

[¶] School of Electrical and Electronic Engineering, Hanoi University of Science and Technology

[**] Faculty of Electrical and Electronics Engineering, Hanoi Open University

[††] Faculty of Electrical and Electronics Engineering, Hanoi Open University

[‡‡] School of Electrical and Electronic Engineering, Hanoi University of Science and Technology