**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC PHENIKAA**

A logo of a university

AI-generated content may be incorrect.

**BÁO CÁO BÀI TẬP LỚN THỊ GIÁC MÁY TÍNH**

**ĐỀ TÀI : NHẬN DIỆN BIỂN SỐ XE VIỆT NAM**

**Sinh viên thực hiện :**

**- Đoàn Văn Hiệp (MSSV: 23010739)**

**- Phạm Quỳnh Chi (MSSV:** **23014680)**

**Lớp tín chỉ: THỊ GIÁC MÁY TÍNH 1-3-24 (N04)**

**Lớp hành chính: KHMT-TN,CNTT-VJ**

**TÓM TẮT ĐỒ ÁN**

**Giới thiệu đề tài: Trong bối cảnh đô thị hiện đại, việc quản lý phương tiện ra vào tại các bãi đỗ xe ngày càng đòi hỏi sự chính xác và tự động hóa cao. Hệ thống kiểm soát thủ công truyền thống không chỉ tốn nhiều nhân lực mà còn tiềm ẩn rủi ro sai sót, giảm hiệu suất vận hành. Nhằm giải quyết bài toán này, đề tài đề xuất xây dựng một hệ thống nhận diện biển số xe tự động bằng cách kết hợp giữa mô hình phát hiện và phân đoạn đối tượng YOLOv8 Segment với mạng nhận dạng ký tự CRNN (Convolutional Recurrent Neural Network).**

**Phương pháp tiếp cận này giúp xác định chính xác vị trí biển số trong ảnh đầu vào, ngay cả khi biển số bị che khuất một phần, chịu ảnh hưởng của điều kiện ánh sáng phức tạp hoặc bị biến dạng do góc chụp. Sau khi phân đoạn, các biển số được chuyển sang mô-đun CRNN để trích xuất và nhận dạng chuỗi ký tự một cách hiệu quả. Hệ thống có thể triển khai thực tế tại các bãi giữ xe thông minh, góp phần tăng cường tính tự động, giảm chi phí vận hành và nâng cao độ tin cậy trong công tác quản lý phương tiện.**

**MỤC LỤC**

**1 Giới thiệu**

**1.1 Tổng quan đề tài**

**1.2 Mục tiêu và phạm vi nghiên cứu**

**1.3 Cấu trúc báo cáo**

**2 Tổng quan về bài toán nhận diện biển số xe**

**2.1 Khái niệm và vai trò của biển số xe**

**2.2 Thách thức trong nhận diện thực tế**

**2.3 Các phương pháp truyền thống và hiện đại**

**2.4 Hướng tiếp cận đề xuất**

**3 YOLOv8\_Segment – Phát hiện và phân đoạn biển số**

**3.1 Tổng quan về YOLO và phiên bản YOLOv8**

**3.2 Kiến trúc YOLOv8\_Segment (Backbone – Neck – Head)**

**3.3 Xử lý ảnh đầu vào và tạo bản đồ phân đoạn**

**3.4 Dữ liệu huấn luyện và kỹ thuật tăng cường**

**3.5 Hàm mất mát và cách tối ưu**

**3.6 Đánh giá mô hình (mAP, IoU, Precision, Recall)**

**3.7 Kết quả thực nghiệm minh hoạ**

**4 CRNN – Nhận dạng ký tự biển số**

**4.1 Cấu trúc mô hình (CNN – RNN – CTC Decoding)**

**4.2 Chuẩn hoá đầu vào từ phân đoạn**

**4.3 Dữ liệu và huấn luyện mô hình CRNN**

**4.4 Đánh giá hiệu suất nhận dạng**

**4.5 Trường hợp đặc biệt (mờ, lệch góc, nhiễu)**

**5 Tích hợp hệ thống tổng thể**

**5.1 Pipeline xử lý từ ảnh tới chuỗi ký tự**

**5.2 Kết nối YOLOv8\_Segment và CRNN**

**5.3 Triển khai thời gian thực**

**5.4 Hạn chế và đề xuất cải tiến**

**6 Kết luận và hướng phát triển**

**7 Tài liệu tham khảo**

**1 GIỚI THIỆU**

**1.1 Tổng quan đề tài**

Trong bối cảnh đô thị hóa nhanh chóng và nhu cầu quản lý phương tiện ngày càng tăng, việc tự động hóa quy trình kiểm soát xe ra vào tại các bãi đỗ xe trở thành một yêu cầu thiết yếu. Hệ thống nhận diện biển số xe (LPR – License Plate Recognition) là một giải pháp công nghệ hiện đại, cho phép phát hiện, phân đoạn và nhận dạng biển số xe thông qua hình ảnh hoặc video, từ đó hỗ trợ quản lý phương tiện một cách hiệu quả, chính xác và tiết kiệm nhân lực.

Truyền thống, các hệ thống LPR dựa trên các kỹ thuật xử lý ảnh cơ bản như phát hiện cạnh, phân ngưỡng và phân đoạn ký tự. Tuy nhiên, các phương pháp này thường gặp khó khăn trong điều kiện ánh sáng yếu, biển số bị mờ hoặc góc chụp không thuận lợi. Sự phát triển mạnh mẽ của trí tuệ nhân tạo, đặc biệt là học sâu (deep learning), đã mở ra hướng tiếp cận mới với độ chính xác cao hơn và khả năng thích ứng tốt hơn với môi trường thực tế.

Trong đề tài này, nhóm nghiên cứu đề xuất xây dựng một hệ thống nhận diện biển số xe dựa trên sự kết hợp giữa hai mô hình học sâu: YOLOv8-Segment để phát hiện và phân đoạn vùng biển số, và CRNN (Convolutional Recurrent Neural Network) để nhận dạng chuỗi ký tự trên biển số. Hệ thống được thiết kế nhằm phục vụ cho các ứng dụng thực tế như quản lý bãi giữ xe, kiểm soát ra vào khu vực giới hạn, hoặc hỗ trợ giám sát an ninh giao thông.

**1.2 Mục tiêu và phạm vi nghiên cứu**

Nghiên cứu này hướng đến việc phát triển một hệ thống nhận dạng biển số xe Việt Nam dựa trên các mô hình học sâu tiên tiến, nhằm hiện đại hóa công tác quản lý phương tiện tại các bãi gửi xe. Mục tiêu chính là xây dựng một giải pháp tự động, có khả năng xử lý hình ảnh biển số trong thời gian thực với độ chính xác cao, kể cả trong điều kiện bất lợi như ánh sáng yếu, góc chụp không chuẩn hoặc biển số bị mờ bẩn. Phạm vi nghiên cứu tập trung vào việc ứng dụng các kiến trúc mạng nơ-ron tích chập (CNN) và kỹ thuật nhận dạng ký tự quang học (OCR) nâng cao để nhận diện biển số xe máy và ô tô phổ biến tại Việt Nam. Hệ thống này không chỉ dễ tích hợp với hạ tầng hiện có, mà còn góp phần giảm thiểu phụ thuộc vào nhân sự vận hành và nâng cao hiệu quả quản lý bãi giữ xe thông minh trong xu hướng đô thị hóa hiện nay.

**1.3 Cấu trúc báo cáo**

**1.3.1 Giới thiệu**

* Lý do chọn đề tài.
* Mục tiêu và phạm vi nghiên cứu.

**1.3.2 Cơ sở lý thuyết**

* Nhận dạng biển số xe.
* Mô hình YOLO: phát hiện biển số.
* Mô hình CRNN: nhận dạng ký tự.

**1.3.3 Hệ thống đề xuất**

* Ảnh đầu vào → YOLO phát hiện và crop biển số.
* Resize + padding ảnh.
* CRNN nhận dạng → chuỗi ký tự biển số.

**1.3.4 Thử nghiệm và đánh giá**

* Độ chính xác.
* Tốc độ xử lý.
* Phân tích kết quả.

**1.3.5 Kết luận và hướng phát triển**

* Tóm tắt kết quả đạt được.
* Đề xuất cải tiến và mở rộng ứng dụng.

**2 TỔNG QUAN VỀ BÀI TOÁN NHẬN DIỆN BIỂN SỐ XE**

**2.1 KHÁI NIỆM VÀ VAI TRÒ CỦA BIỂN SỐ XE**

2.1.1 khái niệm

Ở [Việt Nam](https://vi.wikipedia.org/wiki/Vi%E1%BB%87t_Nam), biển kiểm soát xe cơ giới (hay còn gọi tắt là biển kiểm soát, biển số xe) là [tấm biển gắn trên mỗi xe cơ giới](https://vi.wikipedia.org/wiki/Bi%E1%BB%83n_%C4%91%C4%83ng_k%C3%BD_xe" \o "Biển đăng ký xe), được cơ quan [công an](https://vi.wikipedia.org/wiki/C%C3%B4ng_an_nh%C3%A2n_d%C3%A2n_Vi%E1%BB%87t_Nam" \o "Công an nhân dân Việt Nam) cấp (đối với xe quân sự do Bộ Quốc phòng cấp) khi mua xe mới hoặc chuyển nhượng xe. Biển số xe được làm bằng hợp kim [nhôm](https://vi.wikipedia.org/wiki/Nh%C3%B4m" \o "Nhôm) [sắt](https://vi.wikipedia.org/wiki/S%E1%BA%AFt" \o "Sắt), có dạng [hình chữ nhật](https://vi.wikipedia.org/wiki/H%C3%ACnh_ch%E1%BB%AF_nh%E1%BA%ADt" \o "Hình chữ nhật) hoặc hơi [vuông](https://vi.wikipedia.org/wiki/H%C3%ACnh_vu%C3%B4ng" \o "Hình vuông), trên đó có in [số](https://vi.wikipedia.org/wiki/S%E1%BB%91) và [chữ](https://vi.wikipedia.org/wiki/Ch%E1%BB%AF_vi%E1%BA%BFt" \o "Chữ viết) (biển xe dân sự không dùng các chữ cái I, J, O, Q, W. Chữ R chỉ dùng cho xe rơ-moóc, sơ-mi rơ-moóc) cho biết: Vùng và địa phương quản lý, các con số cụ thể khi tra trên máy tính còn cho biết danh tính người chủ hay đơn vị đã mua nó, thời gian mua nó phục vụ cho công tác an ninh, đặc biệt trên đó còn có hình [Quốc huy Việt Nam](https://vi.wikipedia.org/wiki/Qu%E1%BB%91c_huy_Vi%E1%BB%87t_Nam) dập nổi.

**Tiêu chuẩn về kích thước: Ở mỗi nước thường có tiêu chuẩn về** kích thước nhất định, còn riêng Việt Nam tỉ lệ kích thước giữa các biển số là gần như giống nhau. Biển số xe có 2 loại, kích thước như sau: Loại biển số dài có chiều cao 110 mm, chiều dài 470 mm; loại biển số ngắn có chiều cao 200 mm, chiều dài 280 mm nên ta sẽ giới hạn tỉ lệ cao/rộng là   
3.5 ≤ cao/rộng ≤ 6.5 (biển một hàng) và 0.8 ≤ cao/rộng ≤ 1.5 (biển hai hàng).

**Số lượng kí tự trong biển số xe nằm trong khoảng [7,9]. Chiều cao của chữ và số: 80mm, chiều rộng của chữ và số: 40mm.**

**Từ những đặc điểm trên ta có thể thiết lập nhưng thông số, điều khiển để lọc chọn những đối tương phù hợp mà ta cần.**

2.1.2 Vai trò

Biển số xe là một thành phần không thể thiếu trong hệ thống nhận dạng và quản lý phương tiện giao thông. Về mặt pháp lý, biển số xe đóng vai trò như một mã định danh duy nhất, được cơ quan nhà nước cấp phát nhằm xác lập quyền sở hữu và truy xuất thông tin liên quan đến phương tiện, bao gồm dữ liệu kỹ thuật, chủ xe, và lịch sử đăng kiểm.

Trong lĩnh vực quản lý đô thị và an ninh giao thông, biển số là cơ sở để triển khai các giải pháp giám sát thông minh, như hệ thống thu phí tự động, kiểm soát xe vi phạm, quản lý phương tiện tại các bãi giữ xe, và phát hiện xe bị truy nã. Bên cạnh đó, cấu trúc của biển số xe còn thể hiện thông tin địa lý (mã tỉnh/thành phố), giúp nhận diện nguồn gốc đăng ký phương tiện một cách nhanh chóng và chính xác.

Trong bối cảnh chuyển đổi số và phát triển đô thị thông minh, biển số xe không chỉ là công cụ quản lý truyền thống mà còn là dữ liệu đầu vào quan trọng cho các hệ thống sử dụng trí tuệ nhân tạo và thị giác máy tính — đặc biệt trong các ứng dụng như nhận dạng biển số tự động (ALPR) hay hệ thống bãi đỗ xe thông minh.

**2.2 Thách thức trong nhận diện thực tế**

Trong quá trình triển khai hệ thống nhận dạng biển số xe trong điều kiện thực tế, có nhiều yếu tố ảnh hưởng đến độ chính xác và độ tin cậy của mô hình. Một số thách thức tiêu biểu bao gồm:

* Biến dạng và góc chụp không thuận lợi.Biển số có thể bị nghiêng, méo, hoặc chụp ở góc lệch khiến việc trích xuất ký tự trở nên khó khăn.
* Điều kiện ánh sáng phức tạp hình ảnh có thể bị chói sáng, quá tối hoặc thiếu đồng đều, đặc biệt trong môi trường ngoài trời vào ban đêm hoặc dưới thời tiết xấu.
* Biển số bị che khuất hoặc bẩn bùn đất, vật che chắn (như tay lái xe máy, giá chở hàng...) có thể làm mờ hoặc che mất một phần ký tự, gây lỗi nhận dạng.
* Đa dạng về phông chữ và kiểu dáng biển số. Biển số Việt Nam tồn tại nhiều loại khác nhau (trắng, vàng, xanh, đỏ...) với kích thước và kiểu chữ không hoàn toàn đồng nhất.
* Chuyển động và mờ nhoè Phương tiện di chuyển nhanh hoặc máy ảnh không đủ tốc độ màn trập khiến ảnh bị mờ, ảnh hưởng đến khả năng phát hiện và nhận dạng.
* Giới hạn về dữ liệu huấn luyện Thiếu tập dữ liệu lớn, đa dạng và được gán nhãn đúng là một rào cản trong việc huấn luyện mô hình học sâu hiệu quả.

**2.3 Các phương pháp truyền thống và hiện đại**

🔹 Phương pháp truyền thống

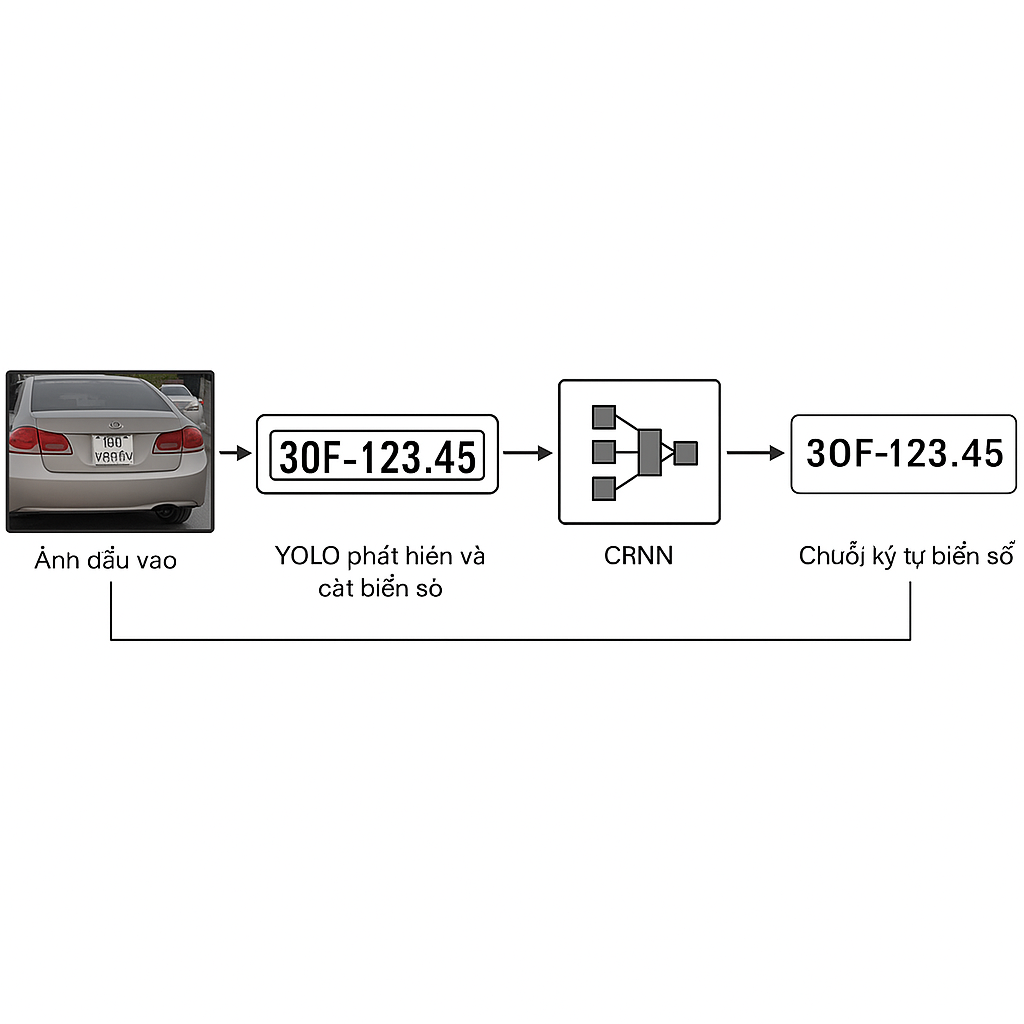
* Tiền xử lý ảnh: Sử dụng các kỹ thuật như chuyển đổi ảnh xám, tăng cường độ tương phản, lọc nhiễu.
* Phát hiện vùng biển số: Dựa vào các đặc trưng hình học như tỷ lệ chiều dài–chiều cao, cạnh sắc nét, hoặc mô hình template.
* Tách ký tự: dùng canny ,findcontour ,thredsholding, morphological.
* Nhận dạng ký tự: Áp dụng các phương pháp học máy cổ điển như SVM, KNN hoặc nhận dạng mẫu dựa trên template. → Ưu điểm: đơn giản, dễ triển khai; →Nhược điểm: kém linh hoạt với ảnh phức tạp, không xử lý tốt các trường hợp nhiễu, biến dạng.

🔹 Phương pháp hiện đại (dựa trên học sâu)

* Phát hiện biển số: Sử dụng mạng CNN hoặc các mô hình như YOLO, Faster R-CNN để tự động phát hiện vùng biển số trong ảnh.
* Nhận dạng ký tự: Áp dụng kiến trúc CRNN kết hợp CTC loss để nhận dạng chuỗi ký tự liền mạch mà không cần tách ký tự.
* Xử lý toàn bộ pipeline bằng end-to-end model: Một số hệ thống tiên tiến sử dụng mô hình dạng Transformer hoặc CNN+RNN tích hợp để xử lý từ đầu vào đến đầu ra chỉ trong một bước. → Ưu điểm: độ chính xác cao, thích ứng tốt với biến dạng và điều kiện thực tế; → Nhược điểm: yêu cầu dữ liệu lớn và tài nguyên tính toán mạnh.

**2.4 Hướng tiếp cận đề xuất**

Đề tài đề xuất một hệ thống nhận dạng biển số xe Việt Nam gồm hai giai đoạn chính. Trước tiên, ảnh đầu vào được xử lý bằng mô hình YOLO để phát hiện và cắt vùng chứa biển số. Tiếp theo, vùng ảnh này được resize + padding để đưa vào mô hình CRNN, giúp nhận diện chuỗi ký tự trên biển số. Hệ thống mang tính chất end-to-end, tự động từ phát hiện đến nhận dạng, phù hợp với yêu cầu thời gian thực và môi trường ứng dụng thực tế.



Hình ảnh minh họa

3. YOLOv8\_Segment – Phát hiện và phân đoạn biển số

**3.1 Tổng quan về YOLO và phiên bản YOLOv8**

**3.1.1 YOLO**

YOLO (You Only Look Once) là một trong những mô hình tiên tiến nhất trong học sâu với nhiều phiên bản phát triển nhất, từ YOLOv1 vào năm 2015 đến năm 2024, đã được lưu hành và phiên bản thứ 11 của YOLO đang được sử dụng.

Bất chấp sự thành công của việc áp dụng YOLO trong số các nhà nghiên cứu và nhà phát triển, vẫn còn những thách thức trong việc

hiểu kiến ​​trúc và các thành phần của nó. Một trở ngại đáng kể là không có các ấn phẩm học thuật và

sơ đồ kiến ​​trúc toàn diện cho một số phiên bản YOLO nhất định, buộc các nhà nghiên cứu và học viên phải

phụ thuộc vào các nguồn thứ cấp và phân tích thực nghiệm để hiểu các mô hình này. Hơn nữa, các bài báo đánh giá hiện có

thường thiếu chiều sâu về các chi tiết cụ thể về kiến ​​trúc và cung cấp những hiểu biết hạn chế. Những thách thức này nhấn mạnh

sự cần thiết phải so sánh kỹ lưỡng để thu hẹp khoảng cách kiến ​​thức. Một so sánh kiến ​​trúc kỹ lưỡng là điều cần thiết để

hiểu chức năng của từng mô hình và nhận ra những cải tiến riêng biệt của chúng.

**3.1.2 YOLOV8**

Dòng sản phẩm YOLO đã phát triển đáng kể kể từ khi ra đời, với mỗi phiên bản đều có những cải tiến và đổi mới quan trọng nhằm cải thiện hiệu suất và hiệu quả

Sau khi được giới thiệu vào năm 2023, YOLOv8 không chỉ cải tiến hiệu suất phát hiện đối tượng nhờ kiến trúc anchor-free, mà còn mở rộng khả năng xử lý với các tác vụ thị giác máy tính khác. Một trong những tính năng nổi bật là YOLOv8 Segment – cho phép mô hình thực hiện phân đoạn thể hiện (instance segmentation).

Khác với phát hiện đối tượng truyền thống chỉ tạo ra hộp giới hạn (bounding box), YOLOv8 Segment còn sinh ra mặt nạ phân vùng (segmentation mask) cho từng đối tượng, giúp xác định chính xác hình dạng và ranh giới của vật thể ở cấp độ pixel. Điều này đặc biệt hữu ích trong các bài toán yêu cầu độ chi tiết cao như xử lý ảnh y tế, xe tự lái, hoặc loại bỏ nền trong ảnh/video.

**3.2 Kiến trúc YOLOv8\_Segment (Backbone – Neck – Head)**

🔷 1. Backbone – Trích xuất đặc trưng

Backbone là thành phần đầu tiên, có nhiệm vụ trích xuất đặc trưng không gian từ ảnh đầu vào thông qua các tầng tích chập sâu.

* Conv-BN-Activation: Tổ hợp lớp tích chập 2D, chuẩn hóa theo batch và hàm kích hoạt (thường dùng SiLU).
* C2f Block (Cross-Stage Partial with Fusion): Khối tích chập cải tiến giúp tăng hiệu quả truyền thông tin và giảm số lượng tham số.
* Downsampling: Thực hiện bằng các lớp Conv với stride=2 để giảm kích thước không gian và tăng chiều sâu đặc trưng.

Backbone tạo ra các đặc trưng ở nhiều cấp độ (multi-scale features), phục vụ cho các tầng sau.

🔷 2. Neck – Kết hợp đặc trưng

Neck là phần trung gian, giúp kết hợp và truyền tải đặc trưng từ nhiều tầng khác nhau nhằm tăng khả năng phát hiện đối tượng ở nhiều kích thước.

* FPN (Feature Pyramid Network): Truyền thông tin từ các tầng sâu lên tầng nông bằng cách upsample và kết hợp đặc trưng.
* PAN (Path Aggregation Network): Truyền thông tin từ tầng nông xuống tầng sâu để tăng tính ngữ cảnh.
* Concat + C2f: Sau khi kết hợp đặc trưng, sử dụng khối C2f để xử lý và làm mịn thông tin.

Neck tạo ra các đặc trưng hợp nhất, giàu thông tin không gian và ngữ nghĩa.

🔷 3. Head – Dự đoán đầu ra

Head là phần cuối cùng, chịu trách nhiệm dự đoán bounding box, phân loại đối tượng và tạo mặt nạ phân đoạn.

* Detection Head:
  + Dự đoán tọa độ hộp giới hạn (x, y, w, h), độ tin cậy (objectness) và xác suất lớp.
  + Sử dụng các lớp Conv tách biệt cho từng tác vụ (decoupled head).
* Segmentation Head:
  + Sinh ra mask coefficients cho từng đối tượng.
  + Kết hợp với prototype masks (được tạo từ một nhánh riêng trong Neck) để tạo ra mặt nạ phân đoạn chi tiết ở cấp độ pixel.

Cơ chế này cho phép mô hình thực hiện phân đoạn thể hiện (instance segmentation) hiệu quả, đồng thời giữ được tốc độ xử lý cao

A diagram of a diagram

AI-generated content may be incorrect.

Hình ảnh mô phỏng yolov8\_segment

A table with numbers and text

AI-generated content may be incorrect.

Bảng thông số model

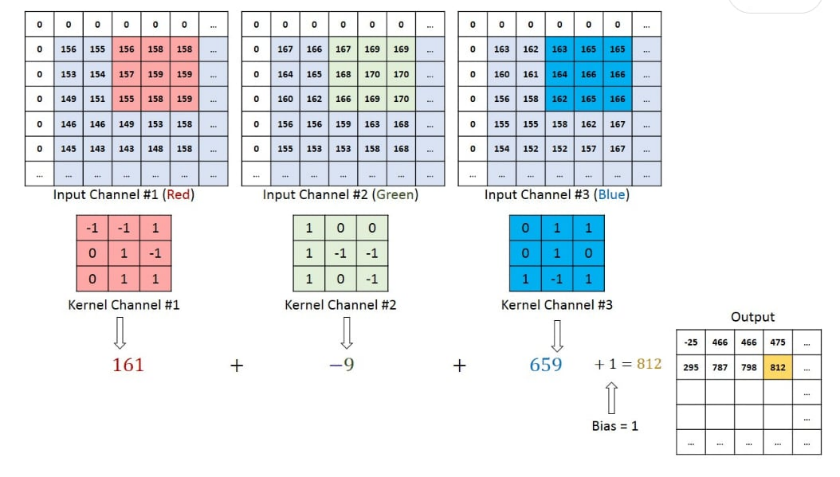
Ví dụ:

Input là ảnh 512\*512\*3 với model là n tức normal.

Bước1:trích xuất đặc trưng từ phần backbone

1. Lớp Conv đầu tiên(512\*512\*3)

* tham số là k=3, s=2,p=1
* ouput=256\*256\*64 (Feature map P1)
* 64: min(64,mc\*w)=min(64,1024\*0.25)=64

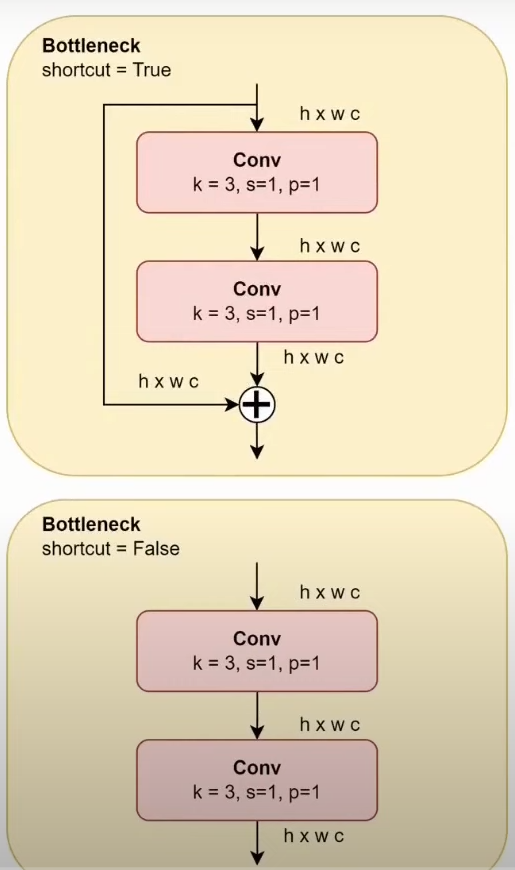


Hình ảnh mô phỏng Conv

1. C2f block

* Tham số:n=1, n=3\*d.
* Shortcut=true
* Output=256\*256\*64

A diagram of a bottleneck

AI-generated content may be incorrect. 

Hình ảnh mô phỏng c2f block

1. Conv 2

* Tham số là k=3,s=2,p=1.
* output= 128\*128\*128.
* 128: min(128,mc\*w)=min(128,1024\*0.25)=128.

1. C2f block

* Tham số : n=2 ,n=6\*d.
* Output :128\*128\*128

1. Conv 3

* Tham số k=3,s=2,p=1.
* output= 64\*64\*256.

1. C2f block

* Tham số :n=2 ,n=6\*d.
* Shortcut =True.
* Output = 64\*64\*256.

1. Conv 4

* Tham số **:** k=3, s=2, p=1
* Output = 32\*32\*512.

1. C2f block

* Tham số: n=1 , n=3\*d
* Shortcut=true.
* Output = 32\*32\*512

1. Conv 5

* Tham số: k=3 s=2 p=1.
* Output 16\*16\*1024

1. C2f block

* Tham số : n=1
* Shortcut=true
* Output = 16\*16\*1024.

1. SPPF

* Mục đích**:** Tổng hợp đặc trưng ở nhiều tỉ lệ khác nhau từ một feature map đầu vào duy nhất. SPPF nhanh hơn SPP truyền thống.
* Output = 16\*16\*1024.

A diagram of a diagram

AI-generated content may be incorrect.

Hình ảnh mô phỏng SPPF Block

Bước 2: NECK: tổng hợp feature- FPN+PAN

1. FPN path ( luồng từ trên xuống dưới)
2. Upsample

* Từ output SPPF Block:16\*16\*1024
* Tăng kích thước input\*2
* Output : 32\*32\*1024

1. Concat (Conv 4 +upsample)

* Input 1 = 32\*32\*512
* Input 2= 32\*32\*1024
* Mục tiêu: kết hợp các feature map có cùng kích thước không gian để tăng số lượng kênh
* Output 32\*32\*(1024+512)

1. C2f block

* Tham số :n=1, shortcut= False
* Input 32\*32\*1536
* Output 32\*32\*512

1. Upsample

* 32\*32\*512
* Out: 64\*64\*512

1. Concat

* In 1:64\*64\*512
* In 2: 64\*64\*256
* Out: 64\*64\*768

1. C2f block

* In : 64\*64\*768
* Tham số n=1 short=false
* Out 64\*64\*256

1. PAN ( lan truyền dưới lên trên)
2. Conv

* Tham số : k =3 s=2 p=1
* In 64\*64\*256
* Out 32\*32\*256

1. Concat

* In 1: 32\*32\*256
* In 2: 32\*32\*512
* Out : 32\*32\*768

1. C2f block

* In 32\*32\*768
* Tham số n=1 short=false
* Out 32\*32\*512

1. Conv

* In 32\*32\*512
* Tham số k=3 s=2 p=1
* Out 16\*16\*512

1. Concat

* In 1 16\*16\*512
* In 2 16\*16\*1024
* Out 16\*16\*1536

1. C2f block

* In 16\*16\*1536
* Tham số n=1 short=false
* Out 16\*16\*1024

Bước 3: HEAD

* Input 1 : 64\*64\*256
* Đi vào phần detect head: dự đoán các bounding box và xác suất lớp
* Phần segment head: dự đoán các hệ số cho mask phân đoạn
* In 2: 32\*32\*512
* Đi vào phần detect head: dự đoán các bounding box và xác suất lớp
* Phần segment head: dự đoán các hệ số cho mask phân đoạn
* In 3: 16\*16\*1024
* Đi vào phần detect head: dự đoán các bounding box và xác suất lớp
* Phần segment head: dự đoán các hệ số cho mask phân đoạn
* Segmentation Process
* **ProtoNet:**

Đây là một mạng con (thường là một tập hợp các lớp tích chập upsampling) được kết nối với một feature map có độ phân giải cao từ Neck (ví dụ, thường là P3\_out hoặc một phiên bản upsample của nó).

**Mục đích:** Tạo ra một tập hợp các **Prototype Masks**. Đây là các mặt nạ cơ bản, có độ phân giải thấp, độc lập với số lượng đối tượng được phát hiện.

**Đầu ra:** 32 Prototype Masks có kích thước 128 x 128. Điều này có nghĩa là ProtoNet tạo ra 32 "mẫu" mask khác nhau mà từ đó các mask cuối cùng sẽ được xây dựng. (Kích thước 128x128 là upsample 2x từ 64x64 của P3\_out, hoặc có thể là 4x từ 32x32 tùy theo cấu hình cụ thể của ProtoNet trong Ultralytics).

* **Mask Assembly (Lắp ráp Mask):**

Đối với mỗi đối tượng được phát hiện (từ Detect Head), Segment Head tương ứng sẽ dự đoán một tập hợp các **Coefficients** (ví dụ: 32 hệ số cho mỗi hộp).

Các hệ số này sau đó được sử dụng để **kết hợp tuyến tính (linear combination)** các Prototype Masks từ ProtoNet. Mỗi hệ số tương ứng với mức độ đóng góp của một Prototype Mask cụ thể vào mask cuối cùng của đối tượng đó.

Công thức cơ bản: Maskfinal​=∑i=132​(Coefficienti​×PrototypeMaski​)

Kết quả cuối cùng được đưa qua hàm **Sigmoid** để tạo ra các giá trị pixel từ 0 đến 1, biểu thị xác suất pixel đó thuộc về đối tượng.

**Đầu ra:** Các **Final Mask** có cùng kích thước với ảnh đầu vào (512x512) hoặc kích thước gần đó (ví dụ, 160x160 hoặc 320x320 tùy theo cấu hình đầu ra của mask trong Ultralytics, sau đó được resize lên kích thước ảnh gốc nếu cần).

A screenshot of a computer

AI-generated content may be incorrect.

Hình ảnh yolov8n\_segment

**3.3 Xử lý ảnh đầu vào và tạo bản đồ phân đoạn**

**3.3.1. Tiền xử lý ảnh đầu vào**

Trước khi đưa vào mô hình, ảnh được xử lý theo các bước sau:

* Resize: Ảnh được thay đổi kích thước về kích thước chuẩn (H,W)(H, W), ví dụ 512×512512 \times 512, đảm bảo đồng nhất đầu vào.
* Chuẩn hóa pixel: Mỗi giá trị pixel được chia cho 255 để đưa về khoảng [0,1][0, 1].
* Chuyển đổi định dạng: Ảnh từ định dạng H×W×3H \times W \times 3 được chuyển thành tensor [B,3,H,W][B, 3, H, W], với BB là batch size.
* Padding (nếu cần): Nếu ảnh không vuông, có thể thêm padding để giữ tỉ lệ khung hình.

**3.3.2. Dự đoán đầu ra từ mô hình**

**YOLOv8 Segment thực hiện hai tác vụ song song:**

**a. Phát hiện đối tượng (Detection Head)**

* Dự đoán các bounding box (x, y, w, h), độ tin cậy (objectness) và xác suất lớp.
* Áp dụng Non-Maximum Suppression (NMS) để loại bỏ các box trùng lặp.

**b. Phân đoạn thể hiện (Segmentation Head)**

**3.3.3. Tạo bản đồ phân đoạn**

Mặt nạ phân đoạn của từng đối tượng được tính bằng cách kết hợp hệ số và prototype:

Trong đó:

* 𝑴ᵢ: mặt nạ phân đoạn của đối tượng thứ i
* 𝑐ᵢⱼ: hệ số mask thứ j của đối tượng i
* 𝑷ⱼ: mặt nạ prototype thứ j
* σ: hàm sigmoid đưa kết quả về khoảng [0, 1]

Sau đó:

* Resize mặt nạ về kích thước ảnh gốc
* Thresholding để tạo bản đồ nhị phân (0: nền, 1: đối tượng)
* Áp dụng mask lên ảnh gốc để hiển thị vùng phân đoạn.



Hình ảnh minh họa

**3.4 Dữ liệu huấn luyện và kỹ thuật tăng cường**

1. **Dữ liệu huấn luyện**

* Hơn 5000 ảnh biển số xe thực tế được gãn nhãn khung biển số xe và chia thành tập train và val với nhãn tương ứng.
* Tầm 100 ảnh trên tập test để kiểm tra hiệu suất mô hình sau khi train mô hình Yolo.
* “Hạn chế: Dataset có tính tổng quát cao hơn thực tế, do đó trong điều kiện sản phẩm chính thức triển khai, nên fine-tune mô hình thêm với dữ liệu từ đúng góc camera cố định tại bãi xe để tăng hiệu quả.”

1. **Kỹ thuật tăng cường**

A screenshot of a computer

AI-generated content may be incorrect.

* **Ý nghĩa**

 **Tăng độ đa dạng của dữ liệu**: Giúp mô hình học được các biến thể khác nhau của đối tượng (góc nhìn, ánh sáng, vị trí…)

 **Giảm overfitting**: Tránh việc mô hình ghi nhớ dữ liệu huấn luyện thay vì học đặc trưng tổng quát

 **Tăng khả năng khái quát**: Mô hình hoạt động tốt hơn trên dữ liệu chưa từng thấy

**3.5 Hàm mất mát và cách tối ưu**

Trong quá trình huấn luyện mô hình YOLOv8 Segment, các hàm mất mát (loss functions) đóng vai trò then chốt trong việc đo lường sai lệch giữa dự đoán của mô hình và giá trị thực tế. YOLOv8 sử dụng kiến trúc head tách biệt (decoupled head), cho phép tối ưu hóa riêng biệt cho từng nhiệm vụ: phát hiện đối tượng và phân đoạn thể hiện.

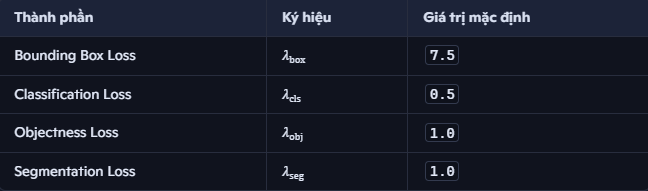
A screenshot of a black background with white text

AI-generated content may be incorrect.

1. **Hàm loss total**

**A white text on a black background

AI-generated content may be incorrect.**

****

1. **Cách tính các hàm loss**

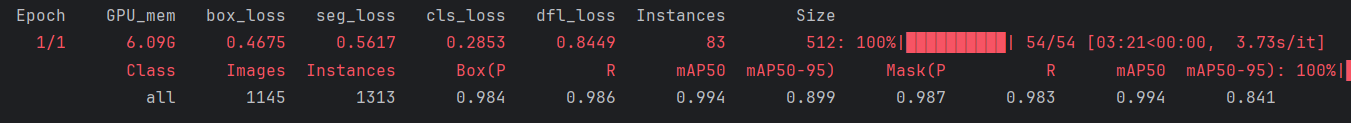
**A screenshot of a computer

AI-generated content may be incorrect.**

**Cách tối ưu hóa**

* **Tổng hợp hàm mất mát**: Tổng loss là tổ hợp tuyến tính của các thành phần trên, với trọng số được điều chỉnh để cân bằng giữa các nhiệm vụ.
* **Tối ưu hóa bằng SGD hoặc Adam**: YOLOv8 sử dụng các thuật toán tối ưu như **SGD** hoặc **AdamW** để cập nhật trọng số mô hình dựa trên gradient của hàm mất mát.
* **Warmup và Cosine LR Scheduler**: Trong giai đoạn đầu huấn luyện, tốc độ học (learning rate) được tăng dần (warmup), sau đó giảm theo hàm cosine để ổn định quá trình học.
* **Early Stopping và EMA**: Mô hình sử dụng kỹ thuật **Exponential Moving Average (EMA)** để lưu giữ các trọng số tốt nhất và tránh overfitting

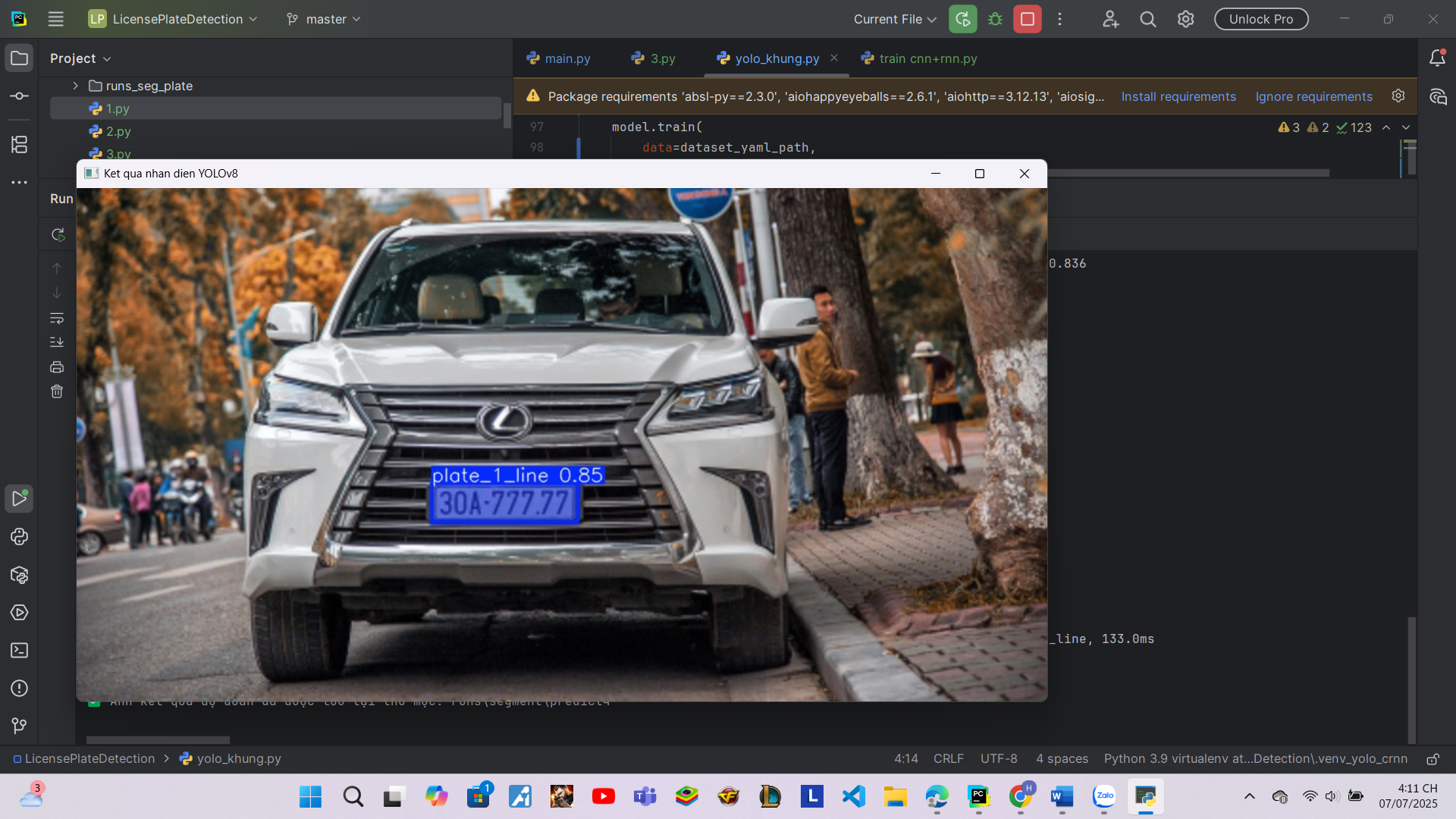
**3.6 Đánh giá mô hình (mAP, IoU, Precision, Recall)**



Hình ảnh train 1 epouch

**3.7 Kết quả thực nghiệm minh hoạ**





Hình ảnh trước và sau khi qua Yolo

* Mô hình YOLOv8 Segment sau huấn luyện đã cho kết quả nhận diện và phân đoạn biển số chính xác trên ảnh thực tế, kể cả trong điều kiện có nhiễu nền. Kết quả minh hoạ cho thấy mô hình không chỉ xác định được vị trí biển số bằng hộp giới hạn mà còn phân tách chính xác vùng biển số qua mặt nạ phân đoạn, chứng tỏ hiệu quả cao khi ứng dụng vào bài toán nhận diện biển số trong môi trường thực tế.

**4 CRNN – Nhận dạng ký tự biển số**

1. **Cấu trúc mô hình (CNN – RNN – CTC Decoding)**

A diagram of a structure

AI-generated content may be incorrect.

Hình ảnh cấu trúc mô hình crnn

1. **Chuẩn hoá đầu vào từ phân đoạn**

Sau khi thực hiện phân đoạn ảnh và xác định được vùng chứa biển số thông qua mô hình YOLOv8 Segment, bước tiếp theo là chuẩn hoá vùng này để đưa vào mô hình nhận dạng ký tự (CRNN). Việc chuẩn hoá giúp đảm bảo độ đồng nhất về kích thước, định dạng và đặc trưng hình ảnh cho các bước nhận dạng phía sau.

* **Cắt vùng biển số**: Sử dụng bounding box từ đầu ra YOLO để cắt ảnh chỉ chứa phần biển số.
* **Chuyển ảnh về grayscale**: Giảm chiều kênh từ RGB còn 1 kênh (đen trắng) nhằm đơn giản hoá thông tin đầu vào.
* **Resize ảnh**: Chuẩn hoá toàn bộ ảnh về kích thước cố định (ví dụ: 256×64) nhằm đảm bảo tính nhất quán với mô hình CRNN.
* **Chuyển tensor**: Ảnh được chuyển thành tensor với định dạng phù hợp (C×H×W), giá trị pixel được chuẩn hoá về khoảng [0, 1].

1. **Dữ liệu và huấn luyện mô hình CRNN**
2. **Dữ liệu**

* Dữ liệu là các ảnh gốc khung chứa biển số xe được cắt từ yolo và được gán nhãn chuỗi text biển số xe thủ công + bán tự động
* Hơn 1000 ảnh đc gán nhãn thủ công bằng scrip code gán nhán tự động bằng cách duyệt qua các ảnh trong file ảnh và đánh text thủ công và lưu lại dưới dạng .txt tương ứng với ảnh với cấu trúc .txt là tên file/tên ảnh/chuỗi text
* Để tăng độ đa dạng của data ta dùng scrip để tạo ra tầm 1000 ảnh giả giống font ảnh biển số xe Việt Nam
* Data được chia thành các tập train và val kèm file label tương ứng dưới dạng .txt

1. Huấn luyện mô hình CRNN

* Các tham số huấn luyện
* Kích thước ảnh đầu vào là 64\*256\*1.
* Max\_text=12: độ dài tối đa của chuỗi text
* Epochs: số lần mô hình được học.
* Batch-size: kích thước batch.
* LR: tốc độ học
* CTC loss : dùng để xử lý chuỗi text dài ngắn khác nhau.
* Dropout :tránh overfitting
* Huấn luyện mô hình

1. Đọc danh sách nhãn từ file train\_rec\_labels.txt, loại bỏ những nhãn không hợp lệ hoặc không nằm trong CHAR\_LIST.
2. Biến đổi ảnh: resize → grayscale → normalize
3. Mã hóa chuỗi ký tự thành chuỗi chỉ số bằng char\_to\_int
4. Ảnh đưa qua CNN → trích đặc trưng không gian
5. Flatten & Linear → biến đổi về dạng phù hợp cho RNN
6. RNN 2 tầng, bidirectional → sinh đặc trưng chuỗi thời gian
7. Lớp fully-connected + log softmax → đầu ra CTC
8. Tính loss với nn.CTCLoss
9. Tối ưu bằng Adam, cập nhật theo từng batch
10. Dự đoán trên tập validation
11. Decode kết quả ra chuỗi ký tự bằng thuật toán “CTC decoding” đơn giản
12. So sánh với ground truth → tính **CER (Character Error Rate)**:
    * + - 1. CER=số ký tự sai/ số ký tự đúng trong gt
13. Lưu mô hình tốt nhất nếu CER thấp hơn trước

A graph with red lines

AI-generated content may be incorrect.

Biểu đồ so sánh các tham số

* Nhận xét

A screenshot of a computer screen

AI-generated content may be incorrect.

* Learning Rate (LR):

 0.0005 (đường đỏ) là mức tối ưu → hội tụ nhanh, đạt CER thấp

 0.001 hội tụ nhanh lúc đầu nhưng bị dao động

 0.0001 hội tụ rất chậm → cần nhiều epoch hơn

* Dropout:

 0.25 (đỏ, xanh dương): giữ được độ chính xác và giảm overfitting

 0.5 (cam): mô hình mất thông tin → CER gần như không thay đổi

* Batch \_size:

 32 cho hiệu quả tốt nhất

 64 khiến CER giảm chậm hơn → có thể do gradient ít ổn định hoặc mô hình khó cập nhật

* Kết luận tổng hợp

Từ biểu đồ so sánh các cấu hình huấn luyện, có thể thấy rằng hiệu quả của mô hình CRNN không chỉ phụ thuộc vào kiến trúc mạng mà còn bị ảnh hưởng mạnh mẽ bởi các siêu tham số như tốc độ học (learning rate), dropout, kích thước batch và số tầng LSTM. Trong quá trình nghiên cứu, mình đã tiến hành thử nghiệm, đối sánh và phân tích chi tiết từng yếu tố, qua đó rút ra được cấu hình tối ưu giúp mô hình đạt độ chính xác nhận dạng cao và hội tụ ổn định hơn.

Việc lựa chọn lr=0.0005, batch\_size=32, dropout=0.25, kết hợp LSTM 2 tầng và bidirectional không chỉ dựa vào giả thuyết học thuật, mà còn được chứng minh rõ ràng thông qua thực nghiệm và theo dõi chỉ số CER. Những thử nghiệm này giúp mình không chỉ hiểu rõ tác động của từng yếu tố riêng lẻ, mà còn nhìn nhận được tương tác giữa các tham số trong toàn bộ pipeline huấn luyện mô hình nhận dạng chuỗi. Đây chính là tiền đề giúp tối ưu mô hình hiệu quả hơn và góp phần khẳng định tính ứng dụng thực tiễn của giải pháp.

1. **Đánh giá hiệu suất nhận dạng**

A screenshot of a computer

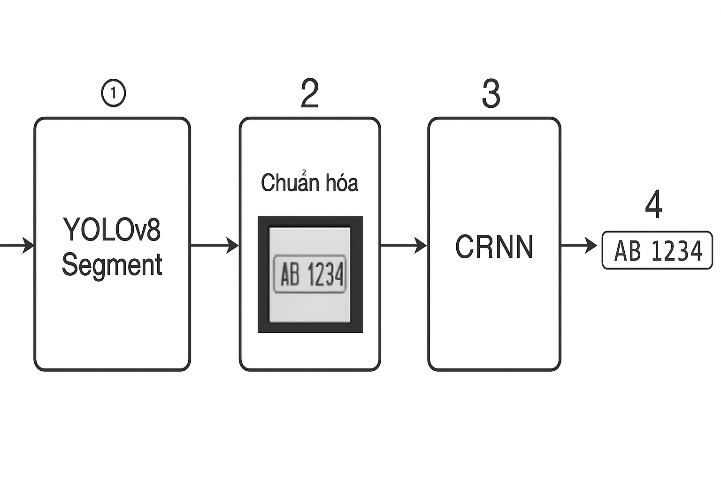
AI-generated content may be incorrect.

* Hình ảnh mô hình CER tốt nhất được lưu với sai số là 8.9%

1. Trường hợp đặc biệt
   * + Do mô hình được train trên dataset ít kém đa dạng nên không thể tổng quát hóa cao khiến cho khi gặp những trường hợp khó khăn trong thực tế như là biển bẩn nhiễu góc chụp k chính diện... khiến cho mô hình nhận dạng ký tự bị sai lệch kém chất lượng hoặc là chất lượng nhận dạng không như mong muốn
     + Giải pháp cho những trường hợp trên là tăng độ đa dạng của tập data huấn luyện kèm theo các hàm tăng cường dữ liệu ảnh như xoay ,độ tăng phản ,nhiễu, .... Cách thức tốt nhất là làm cho data để train rộng lớn và phong phú các trường hợp để mô hình có thể học được nhiều hơn khả năng tổng quát hóa sẽ cao hơn

**5 Tích hợp hệ thống tổng thể**

1. **Pipeline xử lý từ ảnh tới chuỗi ký tự**

****

Hình ảnh cấu trúc pipeline

1. **Kết nối YOLOv8\_Segment và CRNN**

Hệ thống nhận dạng biển số được tổ chức theo mô hình hai giai đoạn:

* YOLOv8 Segment: Phát hiện và phân đoạn biển số trong ảnh đầu vào, cung cấp bounding box và mặt nạ (mask) cho vùng chứa biển số.
* CRNN: Thực hiện nhận dạng ký tự từ ảnh biển số đã được cắt và chuẩn hoá.

Cơ chế kết nối:

* YOLOv8 đưa ra kết quả là vị trí bbox biển số → ảnh gốc sẽ được crop vùng biển số từ bbox này.
* Ảnh sau khi crop sẽ được chuyển sang ảnh xám, resize–pad về kích thước chuẩn (256×64), rồi chuyển thành tensor.
* Tensor đầu vào sau chuẩn hoá sẽ được truyền vào mô hình CRNN để thực hiện nhận dạng chuỗi ký tự.

→ Sự kết hợp này tạo thành một pipeline thống nhất: phát hiện (YOLO) → nhận dạng (CRNN), đảm bảo hệ thống hoạt động tự động và hiệu quả trên ảnh thực tế.

1. **Triển khai thời gian thực**

Hệ thống nhận dạng biển số được thiết kế sẵn sàng để triển khai theo thời gian thực, với luồng xử lý đầu vào từ camera hoặc tập ảnh trực tiếp. Cấu trúc hai tầng (YOLOv8 Segment → CRNN) cho phép phân đoạn và nhận dạng biển số theo thời gian thực nhờ tốc độ xử lý nhanh và độ chính xác cao.

Luồng xử lý:

* Nhận ảnh đầu vào: từ camera giám sát, livestream hoặc chọn file ảnh trực tiếp.
* YOLOv8 Segment: phát hiện vùng biển số và xuất bounding box.
* Crop + chuẩn hoá: cắt ảnh biển số → chuyển ảnh xám → resize + pad → tensor.
* CRNN: nhận dạng ký tự từ ảnh đã chuẩn hóa và xuất chuỗi biển số.
* Hiển thị kết quả trực tiếp trên ảnh: vẽ khung + text dự đoán.

Các đặc điểm hỗ trợ real-time:

* Mô hình CRNN nhẹ, inference nhanh với GPU hoặc CPU.
* Tích hợp GUI bằng tkinter cho phép chọn file ảnh dễ dàng.
* Tùy chọn mở rộng: đọc ảnh từ webcam, hiển thị real-time bằng OpenCV (cv2.VideoCapture()).

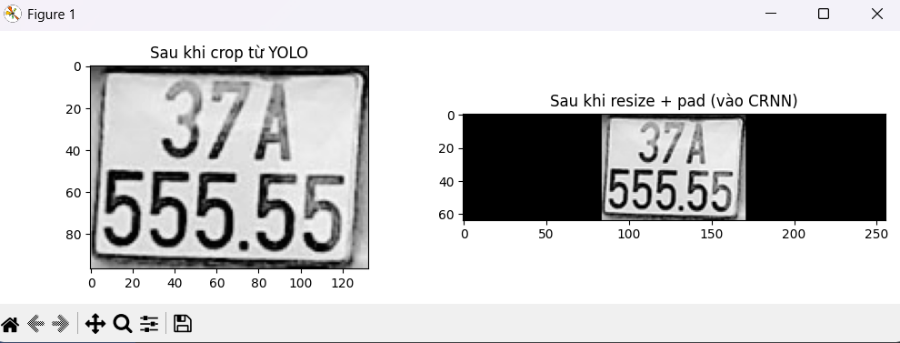
1. **Hạn chế và đề xuất hướng cải tiến mô hình**

**Hạn chế:**

Một trong những thách thức đáng chú ý trong hệ thống là bước resize + pad ảnh biển số trước khi đưa vào mô hình CRNN. Khi ảnh gốc sau khi crop từ YOLO có tỷ lệ quá nhỏ hoặc hẹp (về chiều ngang hoặc chiều dọc), quá trình resize–pad theo kích thước cố định 256×64256 × 64 khiến:

* Phần thông tin ảnh chiếm diện tích nhỏ, còn lại là các viền đen padding chiếm ưu thế.
* Điều này làm giảm chất lượng đặc trưng không gian thu được từ CNN backbone.
* Dẫn tới việc mô hình CRNN khó nhận dạng chính xác do đặc trưng đầu vào không rõ ràng hoặc bị mất tỉ lệ thực.

Ảnh minh họa dưới đây cho thấy rõ vấn đề:



Trái: ảnh gốc sau khi crop từ YOLO Phải: ảnh sau khi resize + pad — phần thông tin bị thu nhỏ, nhiều viền đen.

**Trải nghiệm cải tiến thực tế:**

Mình đã tiến hành thử nghiệm với hai hướng xử lý ảnh đầu vào:

* Cắt rộng bbox từ YOLO hơn bình thường để giữ tỉ lệ hình ảnh tốt hơn khi resize
* Resize mà không pad (chấp nhận ảnh bị méo nhẹ) để giảm vùng đệm đen Tuy nhiên, cả hai cách tiếp cận này đều không đem lại cải thiện rõ rệt — thậm chí còn khiến chất lượng nhận dạng giảm, do mô hình khó tổng quát hóa được ảnh đầu vào không đồng nhất.

Giải pháp khả thi hơn là huấn luyện mô hình CRNN với tập dữ liệu đầu vào đa dạng, bao gồm các TH đã resize + pad lệch tỉ lệ hoặc bị méo. Tuy nhiên, vì giới hạn thời gian và tài nguyên, mình chưa thể triển khai phần mở rộng tập dữ liệu này.

**Đề xuất cải tiến mô hình:**

* Áp dụng một mô-đun Attention hoặc Spatial Transformer giúp mô hình tự động tập trung vào vùng chứa ký tự, bỏ qua vùng pad.
* Nghiên cứu các kỹ thuật adaptive pooling để cho phép mô hình xử lý ảnh đầu vào với chiều kích thước động mà không cần pad cứng về 256×64.
* Nếu có thời gian mở rộng sau đồ án, có thể xây dựng tập dữ liệu chuyên biệt với nhiều kiểu crop + pad → từ đó mô hình học cách tự thích ứng.

**Mở rộng nếu có thêm thời gian và tài nguyên**

Với thời gian nghiên cứu giới hạn, việc mở rộng tập dữ liệu huấn luyện với các trường hợp ảnh bị padding nhiều chưa được thực hiện đầy đủ. Tuy nhiên, nếu có thêm nguồn lực, mình sẽ triển khai các hướng sau nhằm khắc phục triệt để hạn chế:

1. Xây dựng tập dữ liệu tổng hợp (synthetic dataset):
   * Tự động sinh ảnh biển số với nhiều tỉ lệ, padding, xoay, bóp méo khác nhau.
   * Mô phỏng thực tế: biển bị chụp lệch góc, đèn xe phản chiếu, biển mờ do tốc độ cao...
2. Huấn luyện CRNN đa domain đầu vào:
   * Kết hợp dữ liệu gốc và dữ liệu được padding/méo lệch → mô hình học được tính kháng biến dạng tốt hơn.
   * Có thể áp dụng pretraining trên dữ liệu tổng quát, sau đó fine-tune trên dữ liệu thật từ camera thực tế.
3. Tích hợp module tự động căn giữa vùng ký tự:
   * Áp dụng kỹ thuật như Spatial Transformer Networks (STN) hoặc attention trước CRNN giúp mô hình tập trung vào vùng chứa ký tự, bỏ qua viền đen.
4. Thử nghiệm kiến trúc Adaptive Input:
   * Xây dựng CRNN đầu vào linh hoạt, không cần resize cứng → sử dụng adaptive pooling hoặc biến thể CNN theo chiều rộng (width-agnostic CNN) để duy trì tỉ lệ ảnh thực.

**6. Kết luận và hướng phát triển**

Trong đề tài này, mình đã xây dựng thành công một hệ thống nhận dạng biển số xe tự động gồm hai giai đoạn: phân đoạn biển số bằng YOLOv8 Segment và nhận dạng chuỗi ký tự bằng mô hình CRNN. Quá trình triển khai bao gồm:

* Huấn luyện mô hình CRNN từ dữ liệu ảnh biển số đã cắt, qua bước xử lý chuẩn hoá và padding.
* Tích hợp pipeline nhận diện từ ảnh đầu vào → phát hiện → cắt ảnh → chuẩn hoá → nhận dạng ký tự.
* Thực nghiệm đánh giá ảnh hưởng của các siêu tham số như learning rate, batch size, dropout,... và từ đó xác định cấu hình tối ưu giúp mô hình đạt độ chính xác tốt nhất.
* Phân tích các hạn chế kỹ thuật (như ảnh hưởng của padding trong resize ảnh nhỏ) và thử nghiệm một số hướng cải thiện cụ thể.

Hệ thống đã cho thấy khả năng hoạt động tốt trên ảnh tĩnh và sẵn sàng để mở rộng sang các ứng dụng thời gian thực như xử lý camera giám sát, trạm thu phí không dừng hoặc quản lý bãi xe.

**Hướng phát triển trong tương lai:**

* Mở rộng và đa dạng hoá tập dữ liệu huấn luyện để nâng cao khả năng tổng quát hóa của mô hình với các trường hợp thực tế như biển số mờ, lệch, thiếu sáng,...
* Tích hợp module attention hoặc spatial transformer để mô hình CRNN có thể tự động tập trung vào vùng chứa ký tự, giảm ảnh hưởng của padding.
* Phát triển module đọc từ video hoặc camera thời gian thực, xử lý tuần tự từng khung hình và tích hợp đầu ra vào hệ thống quản lý biển số.
* Thử nghiệm các kiến trúc nhận dạng hiện đại hơn như TRBA, ABINet,... để cải thiện tốc độ hội tụ và khả năng kháng biến dạng.

**7. Tài liệu tham khảo**

* <https://arxiv.org/pdf/2501.13400>
* https://thigiacmaytinh.com/phan-doan-doi-tuong-voi-yolov8-segmentation/
* <https://viso.ai/deep-learning/yolov8-guide/>
* <https://www.kaggle.com/code/pranavanand24/applied-deep-learning-nn-cnn-rnn/notebook#Recurrent-Neural-Network>
* <https://www.youtube.com/watch?v=HQXhDO7COj8>
* <https://www.kaggle.com/code/pkdarabi/traffic-signs-detection-using-yolov8>
* Các chatbot AI.
* <https://www.kaggle.com/datasets/duydieunguyen/licenseplates/data>
* YOLOv8 Official repo & whitepaper
* Scene Text Recognition with CRNN