**TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

NGÀNH KHOA HỌC MÁY TÍNH

**---🙠**🕮**🙢---**

Icon

Description automatically generated

**BÁO CÁO ĐỒ ÁN**

**Thị giác máy tính**

***Nhóm 5 -Lớp 67CS1***

**ĐỀ TÀI: NHẬN DIỆN VÀ PHÁT HIỆN KHỐI U**

**TRONG NÃO**

**GVHD: Thái Thị Nguyệt**

**SVTH:** Mai Phạm Lan Anh 0263367

Lê Việt Hùng 0174367

Văn Đức Anh 0279567

Nguyễn Hoài Nam 0055167

***Hà Nội, tháng 5 năm 2025***

**LỜI CẢM ƠN**

Lời đầu tiên, nhóm 5 Đồ án Thị giác máy tính lớp 67CS1 xin gửi lời cảm ơn chân thành và sâu sắc nhất đến cô Thái Thị Nguyệt, giảng viên bộ môn Thị giác máy tính, cô đã luôn tận tâm hướng dẫn và đồng hành cùng nhóm trong suốt quá trình học tập và thực hiện đồ án này. Những kiến thức mà cô truyền đạt không chỉ giúp nhóm nắm bắt được lý thuyết quan trọng mà còn khơi dậy sự sáng tạo và tinh thần học hỏi. Chính nhờ những bài giảng và các ví dụ minh họa thực tế của cô mà nhóm có cơ hội tiếp cận và vận dụng kiếm thức vào bài tập một cách hiệu quả.

Chúng em đặc biệt biết ơn cô vì đã sẵn sàng giải đáp mọi thắc mắc, dù là những chi tiết nhỏ nhất. Sự tận tình và những lời khuyên quý giá của cô đã giúp chúng em hiểu rõ hơn được những kiến thức khúc mắc, khó khăn trong quá trình phân tích, thiết kế và phát triển bài tập lớn này.

Ngoài ra nhóm cũng xin gửi lời cảm ơn đến nhà trường đã tạo điều kiện tốt nhất để chúng em có thể tiếp cận các tài liệu học tập, cơ sở vật chất cũng như môi trường học tập thuận lợi để phát triển bản thân.

Một lần nữa, nhóm 5 xin chân thành cảm ơn và kính chúc cô Thái Thị Nguyệt luôn khỏe mạnh, thành công trong công tác giảng dạy và tiếp tục đốt lửa đam mê của các lớp sinh viên sau này.

Xin chân thành cảm ơn!

**Mục lục**

[I. Giới thiệu về đề tài đồ án 4](#_Toc199206617)

[II. Hướng thực hiện đồ án 4](#_Toc199206618)

[1. Mục tiêu đề ra 4](#_Toc199206619)

[2. Hướng thực hiện 4](#_Toc199206620)

[3. Dữ liệu 4](#_Toc199206621)

[4. Mô hình phù hợp 7](#_Toc199206622)

[a. Tổng quan về YOLO 7](#_Toc199206623)

[b. Tổng quan về Yolov5 8](#_Toc199206624)

[c. Kiến trúc Yolov5 8](#_Toc199206625)

[d. Ưu điểm và nhược điểm của YOLO 9](#_Toc199206626)

[e. Tải Yolov5 9](#_Toc199206627)

[5. Các chỉ số đánh giá mô hình: 10](#_Toc199206628)

[a. Precision (Độ chính xác) 10](#_Toc199206629)

[b. Recall (Độ bao phủ) 10](#_Toc199206630)

[c. F1-Score 10](#_Toc199206631)

[d. Mean Average Precision (mAP) 10](#_Toc199206632)

[e. Intersection over Union (IoU) 11](#_Toc199206633)

[f. Ý nghĩa của các chỉ số trong thực tế 11](#_Toc199206634)

[g. Các yếu tố ảnh hưởng đến chỉ số 11](#_Toc199206635)

[III. Quy trình thực hiện 11](#_Toc199206636)

[1. Chuẩn bị dữ liệu: 11](#_Toc199206637)

[2. Các siêu tham số huấn luyện (Hyperparameters) 12](#_Toc199206638)

[3. Huấn luyện mô hình: 12](#_Toc199206639)

[4. Đánh giá mô hình 15](#_Toc199206640)

[5. Hiển thị kết quả 15](#_Toc199206641)

[6. Xét trên một ảnh hoàn toàn mới 16](#_Toc199206642)

[IV. Đánh giá mô hình 17](#_Toc199206643)

[1. Tổng quan mô hình 17](#_Toc199206644)

[2. Phân tích chi tiết theo từng lớp 17](#_Toc199206645)

[3. Kết luận: 18](#_Toc199206646)

[V. Hiển thị kết quả đầu ra 18](#_Toc199206647)

[VI. Hiển thị kết quả với dữ liệu hoàn toàn mới 20](#_Toc199206648)

[VII. Kết luận 20](#_Toc199206649)

[1. Tổng quan bài toán 20](#_Toc199206650)

[2. Kết quả 21](#_Toc199206651)

[3. Hạn chế 21](#_Toc199206652)

1. Giới thiệu về đề tài đồ án

Hiện nay, ung thư não và các loại khối u trong não là một trong những vấn đề sức khỏe nghiêm trọng đe dọa đến tính mạng và chất lượng sống của con người. Khối u là sự phát triển bất thường của các tế bào trong cơ thể. Các tế bào có thể nhân lên một cách không kiểm soát, tạo thành một khối hoặc một mô bất thường. Khối u có thể xuất hiện ở bất kỳ bộ phận nào trong cơ thể. Những khối u não có nhiều hình dậng và kích cỡ, có thể lành tính hoặc ác tính. Việc phát hiện sớm và chính xác các khối u trong não đóng vai trò quyết định trong việc điều trị, giúp tăng cơ hội sống sót và cải thiện sức khỏe lâu dài cho bệnh nhân. Tuy nhiên, các phương pháp chuẩn đoán hiện nay chủ yếu dựa vào kỹ năng và kinh nghiệm của bác sĩ thường gặp nhiều khó khăn khi phân tích hành loạt ảnh cộng hưởng từ (MRI) với độ phức tạp cao, tốn nhiều thời gian và có thể xảy ra chuẩn đoán sai. Điều này thúc đẩy sự cần thiết của các giải pháp hỗ trợ chuẩn đoán tự động, nhanh chóng và hiệu quả.

Đề tài “Nhận diện và phát hiện khối u trong não” với mục đích xây dựng một mô hình giúp nhận diện và phát hiện khối u trong não từ hình ảnh MRI. Sử dụng tập dữ liệu chứa ảnh MRI để huấn luyện mô hình có thể giúp phân loại hình ảnh khối u hoặc không phải khối u nhằm đưa ra một kết quả với độ chính xác tương đối

1. Hướng thực hiện đồ án
2. Mục tiêu đề ra

* Sử dụng mô hình YOLO để phát hiện và khoanh vùng khối u trên ảnh MRI của não.
* Phát hiện và nhận diện khối u trong ảnh chụp MRI não với các chỉ số tương đối.
* Áp dụng các bộ lọc tiền xử lý dữ liệu để cải thiện các chỉ số của mô hình.
* In ra màn hình các vị trí khối u sau khi đã train mô hình
* Đánh giá hiệu suất mô hình dựa trên các tham số: mAP (mean Average Precision), Precision, Recall.
* Đưa 1 ảnh MRI hoàn toàn mới vào và mô hình sẽ đưa ra dự đoán là khối u đó thuộc loại nào.

1. Hướng thực hiện

Để thực hiện đề tài của đồ án một cách chính xác nhất, phải tìm hiểu về các kiến thức cần sử dụng để áp dụng vào bài một cách hiệu quả.

Tìm tập dữ liệu và tìm hiểu về các mô hình có thể sử dụng, các bộ lọc để tiền xử lý dữ liệu, phần mềm gán nhãn.

Thực hiện đề tài trên Google Colaboratory hay còn gọi là Google Colab để đảm bảo cho việc thực hiện, theo dõi quá trình.

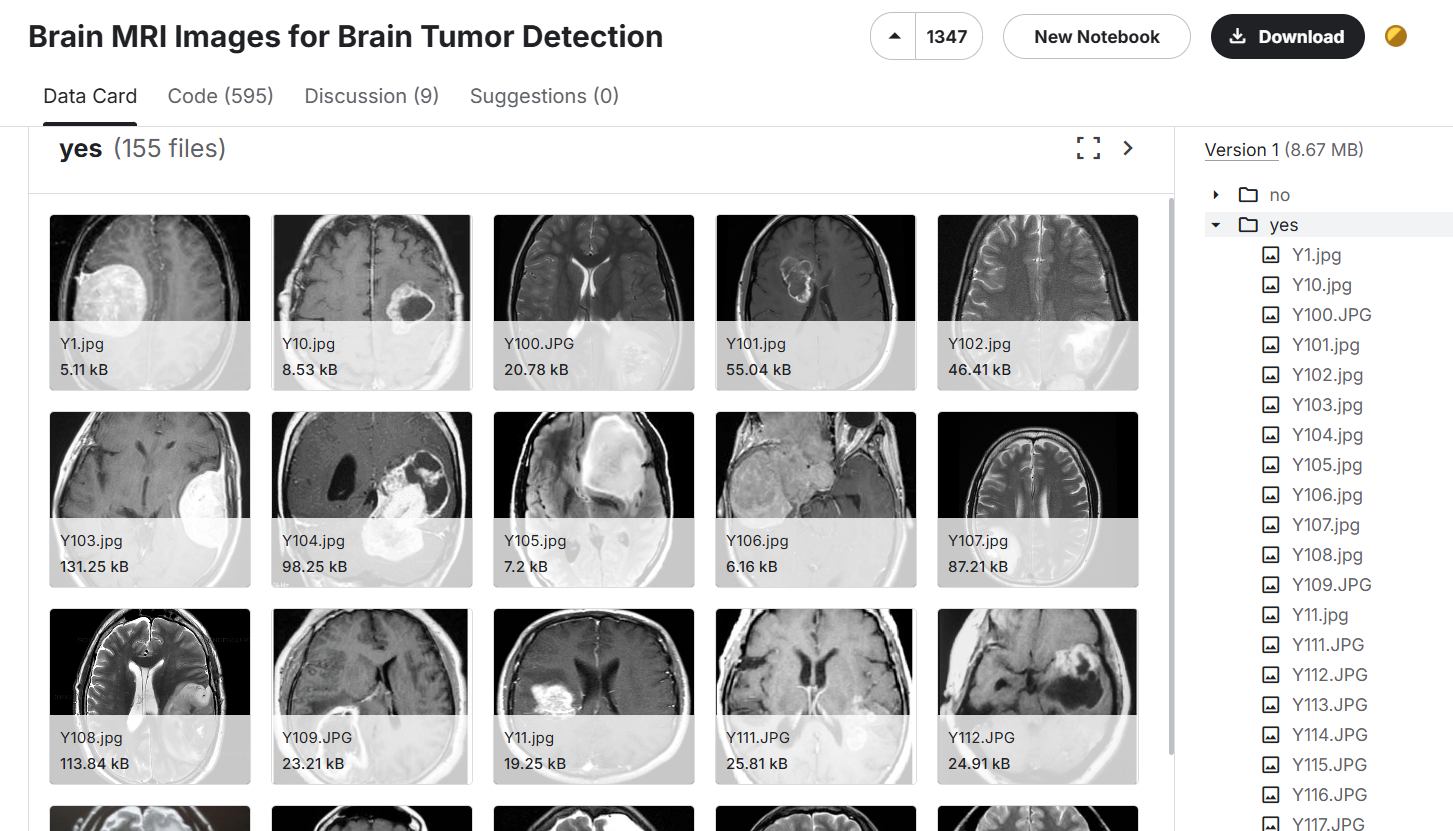
1. Dữ liệu

Dữ liệu trong đề tài sẽ là tập dữ liệu các ảnh MRI não sẽ chứa các khối u và không có khối u. Để tìm dữ liệu, sẽ tìm kiếm trên các nguồn dữ liệu lớn như Kaggle, roboflow,...

Các bộ dữ liệu được chọn sử dụng:

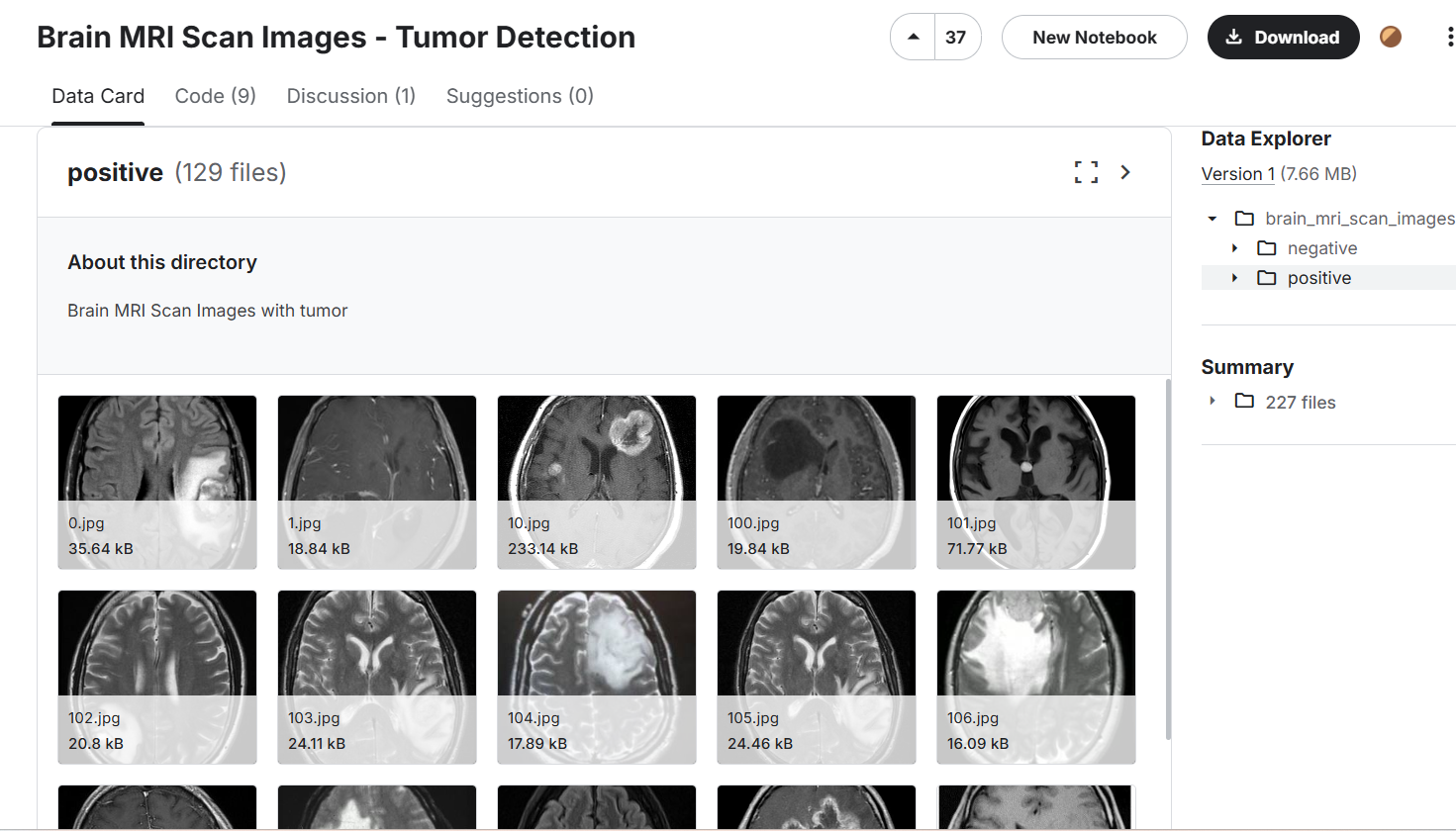
* Brain MRI Images for Brain Tumor Detection:

Link: <https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection/data>

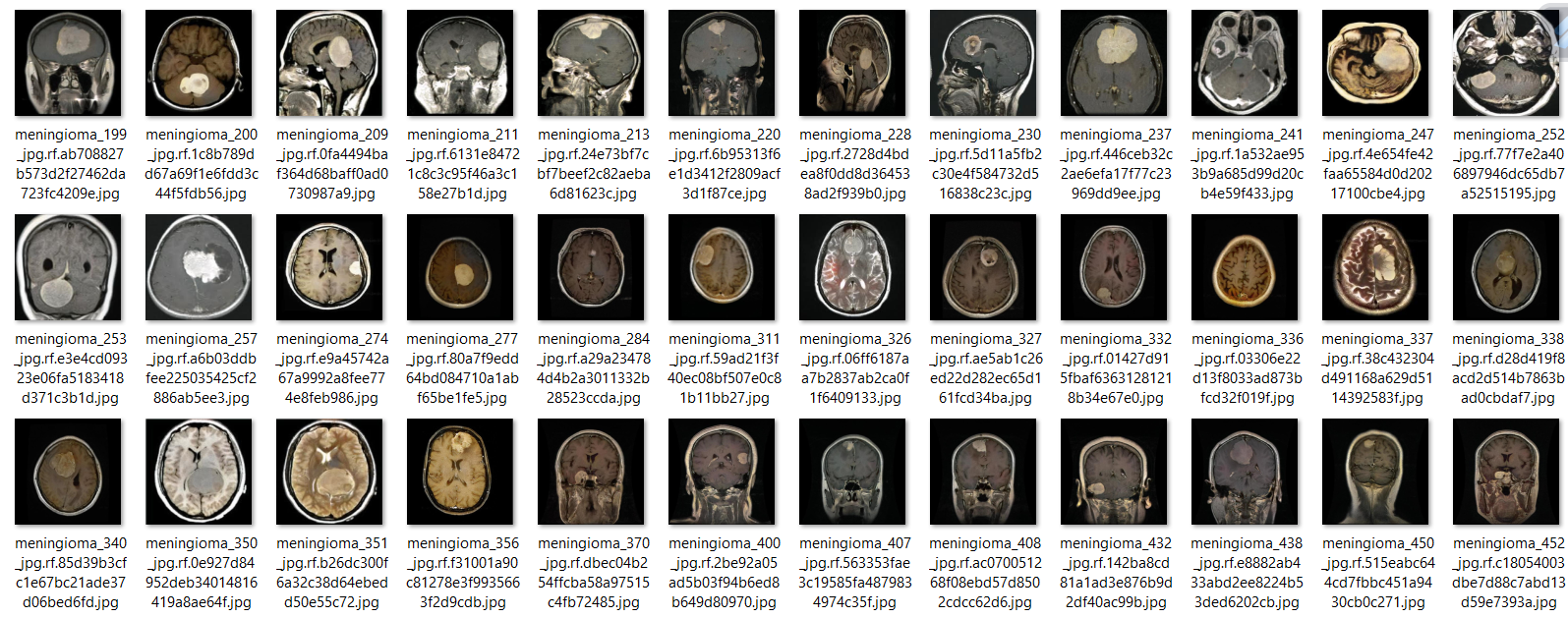


* Brain Tumor MRI:

Link: <https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset>



Sau khi đã sưu tầm đủ dữ liệu, tiến hành đồng bộ dữ liệu:



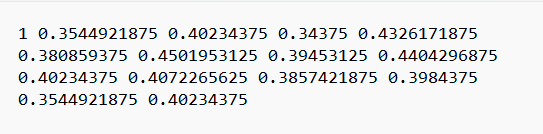
Sau khi chuẩn bị xong dữ liệu, gán nhãn cho từng dữ liệu. Sử dụng phần mềm labelImg để gán nhãn:



Dữ liệu được gán nhãn trên labelImg: khoanh vùng cho từng vùng khối u với từng nhãn liên quan.

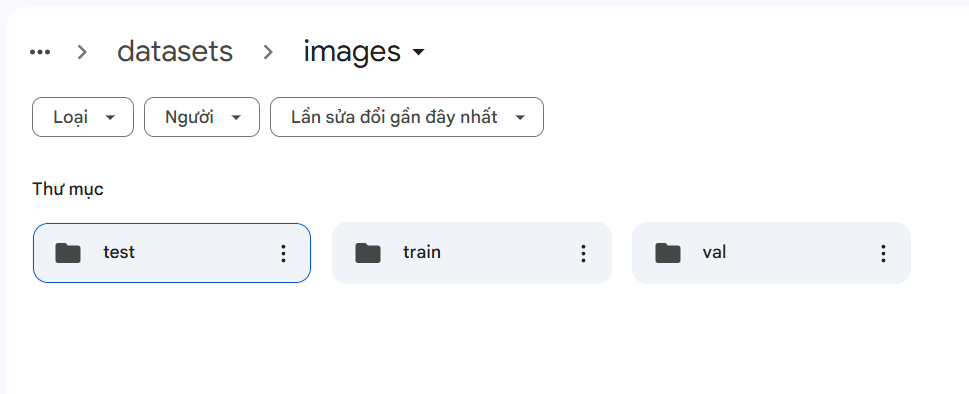


Nhãn sau khi được gán nhãn sẽ được xuất ra file txt để dễ dàng cho việc thực hiện train mô hình.



Dữ liệu đã được gán nhãn xong ta sẽ chia dữ liệu (bao gồm cả ảnh và nhãn) thành 3 tập:

* Tập train: Dùng để huấn luyện mô hình (tập huấn luyện: 70%)
* Tập val: Dùng để đánh giá hiệu suất mô hình (tập kiểm thử: 15%)
* Tập test: Dùng để kiểm tra hiệu suất cuối cùng của mô hình (tập kiểm tra: 15%)



1. Mô hình phù hợp

Sử dụng mô hình Yolov5 để thực hiện train dữ liệu

* YOLO (You Only Look Once) để phát hiện và phân loại khối u não là một phương pháp phổ biến trong các bài toán phát hiện đối tượng (object detection) trong thời gian thực.
* YOLO nổi bật nhờ khả năng xử lý nhanh, hiệu quả và độ chính xác cao, phù hợp cho các ứng dụng như giám sát, nhận diện,…

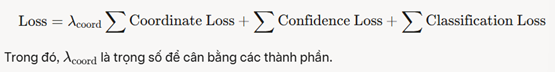
1. Tổng quan về YOLO

YOLO à một mô hình phát hiện đối tượng theo phương pháp "single-shot", nghĩa là nó chỉ cần một lần truyền qua mạng nơ-ron (forward pass) để dự đoán cả vị trí (bounding box) và lớp (class) của các đối tượng trong ảnh. Khác với các phương pháp hai giai đoạn như R-CNN, YOLO xử lý toàn bộ hình ảnh trong một lần duy nhất, giúp tăng tốc độ đáng kể.

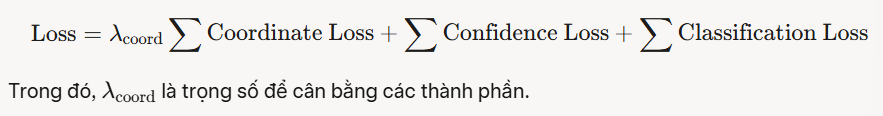
* Nguyên lý cốt lõi của YOLO:
* Chia lưới (Grid Division): YOLO chia hình ảnh đầu vào thành một lưới (grid) kích thước SxS (ví dụ 7x7)
* Dự đoán tại mỗi ô lưới:

- Mỗi ô lưới dự đoán B hộp giới hạn (bounding boxes) với thông tin: tọa độ (x,y), kích thước (w, h) và độ tin cậy (confidence score)

- Đồng thời, mỗi ô dự đoán xác suất thuộc về C lớp (class probabilities)

* Đầu ra: Một tensor chứa thông tin về các hộp giới hạn và lớp đối tượng, sau đó được lọc qua các kỹ thuật Non-Maximum Suppression (NMS) để loại bỏ các dự đoán trùng lặp.
* Công thức Loss Function: YOLO sử dụng hàm mất mát (loss function) kết hớp nhiều thành phần:
* Mất mát tọa độ (Coord inate Loss): Đo lường sai số của tọa độ (x, y) và kích thước (w, h) của hộp giới hạn.
* Mất mát độ tin cậy (Conddidence Loss): Đo lường độ chính xác của confidence score, bao gồm cả trường hợp có đối tượng (objectness) và không có đối tượng.
* Mất mát phân loại (Classification Loss): Đo lường sai số trong dự đoán lớp.

Hàm mất mát tổng quát:



1. Tổng quan về Yolov5

YOLOv5 được phát triển bởi Ultralytics dựa trên framework PyTorch, ra mắt vào năm 2020. Không như các phiên bản trước (YOLOv1–v4) sử dụng Darknet, YOLOv5 được thiết kế để dễ tiếp cận, với mã nguồn mở và tài liệu chi tiết. Nó không phải là một bước đột phá lớn về lý thuyết so với YOLOv4, nhưng được ưa chuộng nhờ tính linh hoạt và hiệu suất.

Các biến thể của Yolov5: YOLOv5 có nhiều kích thước mô hình, phù hợp với các nhu cầu khác nhau về tốc độ và độ chính xác:

* YOLOv5n (nano): Nhỏ nhất, nhanh nhất, phù hợp cho thiết bị nhúng (Raspberry Pi, Jetson).
* YOLOv5s (small): Nhẹ, cân bằng tốc độ và độ chính xác.
* YOLOv5m (medium): Hiệu suất tốt hơn YOLOv5s, nhưng chậm hơn.
* YOLOv5l (large): Độ chính xác cao hơn, nhưng yêu cầu phần cứng mạnh.
* YOLOv5x (extra large): Lớn nhất, chính xác nhất, nhưng chậm nhất.

1. Kiến trúc Yolov5

Yolov5 tuân theo cấu trúc chung của các mô hình YOLO: Backbone, Neck và Head

* Backbone:
* Chức năng: Trích xuất đặc trưng từ hình ảnh.
* Cấu trúc: Sử dụng các khối C3 (Cross Stage Partial), kết hợp với Conv2d, BatchNorm, và SiLU activation.
* Đặc điểm: Nhẹ hơn so với Darknet-53 của YOLOv3/v4, tối ưu hóa cho tốc độ và hiệu suất.
* Neck:
* Chức năng: Tổng hợp đặc trưng từ các tầng khác nhau để phát hiện đối tượng ở nhiều kích thước.
* Kỹ thuật:

- SPPF (Spatial Pyramid Pooling - Fast): Tăng khả năng xử lý đặc trưng đa tỷ lệ.

- FPN (Feature Pyramid Network): Kết hợp đặc trưng từ các tầng có độ phân giải khác nhau.

- PANet (Path Aggregation Network): Cải thiện luồng thông tin.

* Head:
* Chức năng: Dự đoán hộp giới hạn (bounding boxes), độ tin cậy (confidence) và lớp (class)
* Đầu ra: Tensor có dạng S x S x (B x (5 + C)) với

- S: Kích thước lưới (thường là 3 tỷ lệ: 19x19, 38x38, 76x76)

- B: Số hộp giới hạn mỗi ô (thường là 3)

- 5: Tọa độ (x, y, w, h) và confidence score

- C: Số lớp

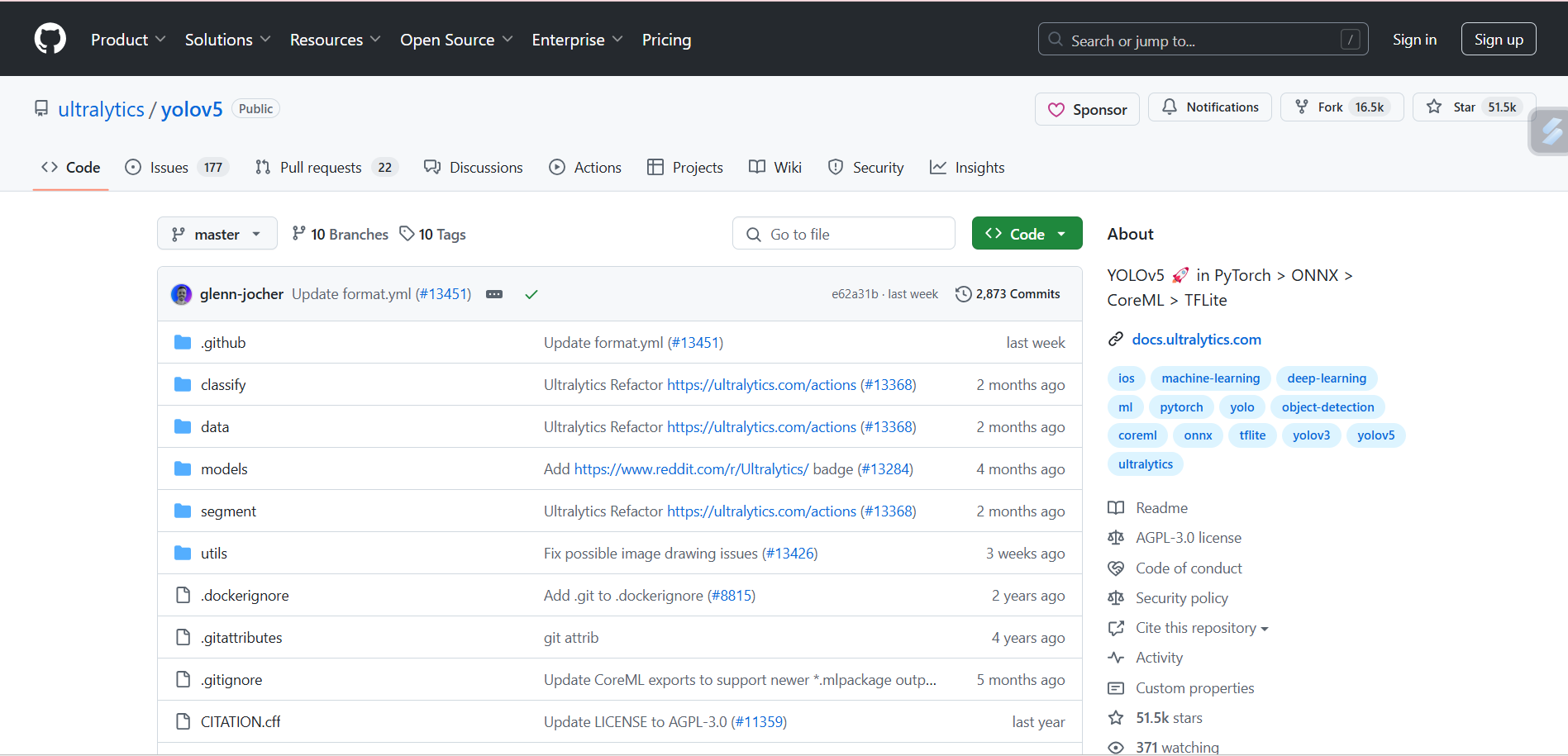
* Anchor Boxes: Yolov5 tự động tối ưu anchor boxes dựa trên tập dữ liệu (auto-anchor optimization), giúp cải thiện khả năng dự đoán hộp giới hạn.

1. Ưu điểm và nhược điểm của YOLO

* Ưu điểm:
* Tốc độ nhanh: Phù hợp cho các ứng dụng thời gian thực (real-time).
* Đơn giản: Chỉ cần một lần truyền qua mạng, giảm độ phức tạp so với các mô hình hai giai đoạn.
* Khả năng tổng quát doanh nghiệp: YOLO có khả năng tổng quát hóa tốt, có thể phát hiện các đối tượng chưa từng thấy trước đó.
* Hỗ trợ đa nhiệm vụ: Các phiên bản mới (như YOLOv8) hỗ trợ phát hiện, phân đoạn, và theo dõi đối tượng.
* Nhược điểm:
* Độ chính xác thấp hơn với đối tượng nhỏ: Do chia lưới cố định, YOLO có thể bỏ sót các đối tượng nhỏ hoặc quá gần nhau.
* Khó khăn trong phát hiện đối tượng đông đúc: Các phiên bản cũ gặp khó khăn khi có nhiều đối tượng trong một ô lưới.
* Yêu cầu phần cứng mạnh: Đặc biệt với các phiên bản mới như YOLOv8, YOLOv9.

1. Tải Yolov5

Tải với đường link: <https://github.com/ultralytics/yolov5>



1. Các chỉ số đánh giá mô hình:
2. Precision (Độ chính xác)

* Định nghĩa: Tỷ lệ giữa số dự đoán đúng (TP) so với tổng số dự đoán tích cực (bao gồm TP và FP)
* Ý nghĩa:
* Đo lường mức độ chính xác của các dự đoán tích cực.
* Precision cao nghĩa là mô hình ít dự đoán sai (ít nhầm các đối tượng không tồn tại).
* Trong YOLOv5: Precision thường được báo cáo trong quá trình đánh giá (validation) và được tính dựa trên ngưỡng IoU (Intersection over Union) cụ thể.

1. Recall (Độ bao phủ)

* Định nghĩa: Tỷ lệ giữa số dự đoán đúng (TP) so với tổng số đối tượng thực tế (bao gồm TP và False Negatives - FN).
* Công thức:
* Ý nghĩa:
* Đo lường khả năng mô hình phát hiện được tất cả các đối tượng thực tế.
* Recall cao nghĩa là mô hình ít bỏ sót đối tượng.
* Trong YOLOv5: Recall cũng được tính trong quá trình đánh giá, thường đi đôi với Precision.

1. F1-Score

* Định nghĩa: Trung bình điều hòa (harmonic mean) giữa Precision và Recall, dùng để cân bằng hai chỉ số này.
* Công thức:
* Ý nghĩa:
* F1-Score cao khi cả Precision và Recall đều tốt.
* Thường được dùng khi cần một chỉ số tổng hợp để đánh giá hiệu suất.
* Trong YOLOv5: F1-Score được báo cáo trong kết quả validation, giúp đánh giá sự cân bằng giữa Precision và Recall.

1. Mean Average Precision (mAP)

* Định nghĩa: Chỉ số phổ biến nhất trong object detection, đo lường trung bình độ chính xác trung bình (Average Precision - AP) trên tất cả các lớp.
* Công thức:
* AP cho một lớp: Tính diện tích dưới đường cong Precision-Recall (PR Curve).
* mAP: Trung bình AP của tất cả các lớp:
* Các biến thể:
* mAP@0.5: Tính AP với ngưỡng IoU = 0.5 (dự đoán được coi là đúng nếu IoU ≥ 0.5).
* mAP@0.5:0.95: Trung bình AP trên các ngưỡng IoU từ 0.5 đến 0.95 (bước 0.05), phản ánh hiệu suất trên nhiều mức độ chồng lấn khác nhau.
* Trong YOLOv5: mAP@0.5 và mAP@0.5:0.95 là các chỉ số chính được báo cáo trong val.py hoặc sau khi huấn luyện.

1. Intersection over Union (IoU)

* Định nghĩa: Đo lường mức độ chồng lấn giữa hộp giới hạn dự đoán (predicted bounding box) và hộp giới hạn thực tế (ground truth).
* Công thức:
* Ý nghĩa:
* IoU cao nghĩa là hộp giới hạn dự đoán khớp tốt với thực tế.
* Thường được dùng làm ngưỡng để xác định True Positive (ví dụ: IoU ≥ 0.5).
* Trong YOLOv5: IoU là cơ sở để tính Precision, Recall, và mAP. Ngưỡng IoU mặc định là 0.5, nhưng có thể điều chỉnh.

1. Ý nghĩa của các chỉ số trong thực tế

* Precision cao, Recall thấp: Mô hình dự đoán chính xác nhưng bỏ sót nhiều đối tượng. Có thể cần giảm ngưỡng confidence hoặc tăng dữ liệu huấn luyện.
* Recall cao, Precision thấp: Mô hình phát hiện nhiều đối tượng nhưng có nhiều dự đoán sai. Có thể cần tăng ngưỡng IoU hoặc cải thiện chất lượng nhãn.
* mAP@0.5:0.95 thấp: Mô hình khó dự đoán chính xác hộp giới hạn ở các ngưỡng IoU cao. Cần tinh chỉnh anchor boxes hoặc sử dụng mô hình lớn hơn (như YOLOv5l).

1. Các yếu tố ảnh hưởng đến chỉ số

* Tập dữ liệu
* Dữ liệu không đủ hoặc không đa dạng làm giảm mAP.
* Nhãn sai hoặc không chính xác ảnh hưởng đến Precision và Recall.
* Kích thước ảnh đầu vào
* Ngưỡng confidence và IoU
* Loại mô hình,…

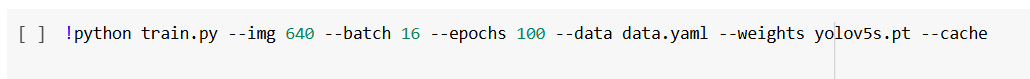
1. Quy trình thực hiện
2. Chuẩn bị dữ liệu:

Chia tập dữ liệu: tập huấn luyện, tập kiểm thử, tập kiểm tra

* Ngẫu nhiên hóa: chia dữ liệu một cách ngẫu nhiên để đảm bảo mỗi tập đầy đủ cho toàn bộ dữ liệu.
* Không trùng lặp: Đảm bảo rằng các mẫu trong tập test không bị trùng tập train hoặc validation.

1. Các siêu tham số huấn luyện (Hyperparameters)

* Learning rate (Tốc độ học):
* Quy định mức độ thay đổi (hoặc bước nhảy) của các trọng số mô hình trong mỗi lần cập nhật dựa trên gradient.
* Giá trị nhỏ → mô hình học chậm nhưng ổn định.
* Giá trị lớn → mô hình học nhanh nhưng dễ vượt qua điểm cực tiểu toàn cục (underfitting) hoặc dao động (không hội tụ).
* Nên bắt đầu với giá trị nhỏ (ví dụ: 0.001 hoặc 0.01).
* Sử dụng các kỹ thuật giảm learning rate trong quá trình huấn luyện (Learning Rate Decay)
* Epochs (Số lần duyệt qua toàn bộ dữ liệu huấn luyện):
* Một epoch là khi mô hình đã duyệt qua toàn bộ tập dữ liệu huấn luyện một lần.
* Đối với đề tài này, sử dụng 100 epochs cho toàn bộ lần train
* Batch Size (Kích thước lô dữ liệu):
* Số lượng mẫu (images) được đưa vào mô hình trong một lần tính toán
* Sử dụng batch 16



1. Huấn luyện mô hình:

Thực hiện huấn luyện mô hình trên Google Colab

* Liên kết với drive – nơi lưu trữ dữ liệu

# liên kết drive

from google.colab import drive

drive.mount('/content/drive')

* Tải thư viện yolov5 và các yêu cầu thư viện:

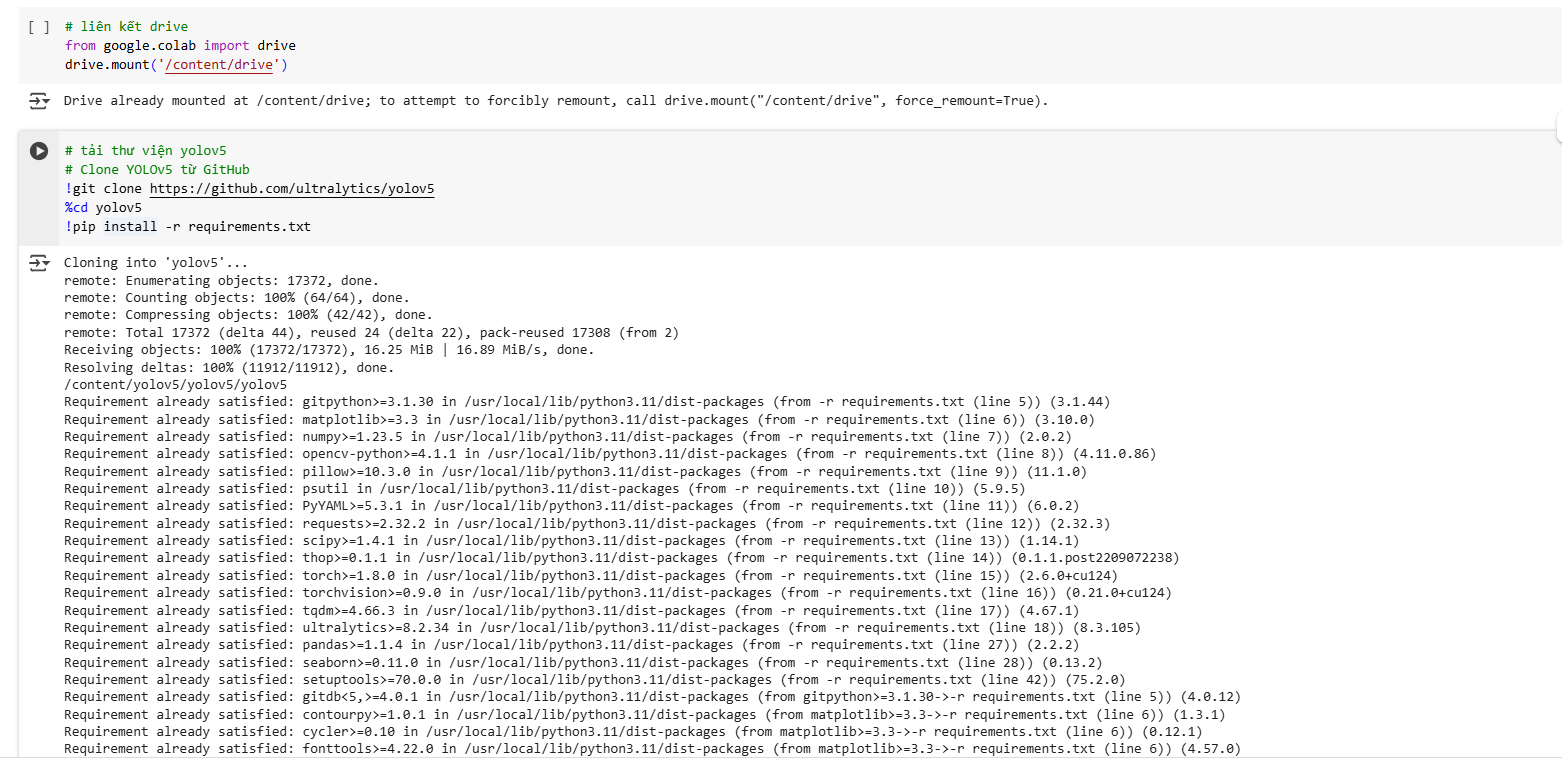
# tải thư viện yolov5

# Clone YOLOv5 từ GitHub

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

%cd yolov5

!pip install -r requirements.txt



* Tạo file data.yaml

with open('data.yaml', 'w') as f:

  f.write('train: /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets/images/train\n')

  f.write('val: /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets/images/val\n')

  f.write('nc: 5\n')

  f.write("names: ['NO\_tumor', 'glioma', 'meningioma', 'pituitary', 'space - occupying lesion -']\n")

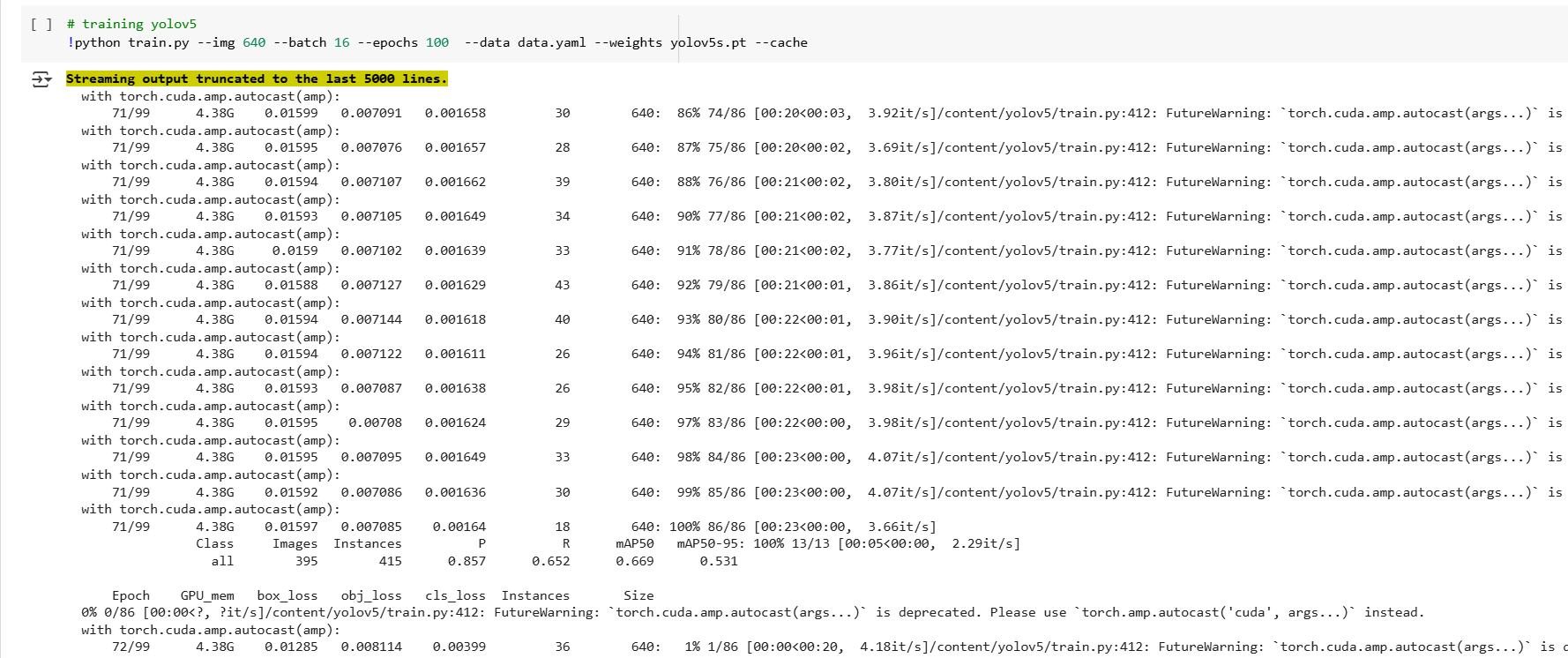
Ở đề tài này sẽ có 5 class ‘NO\_tumor’, ‘glioma’, ‘meningioma’, ‘pituitary’, ‘space-occupying lesion-‘

* NO\_tumor: Lớp này đại diện cho các trường hợp không có khối u trong não.
* glioma: Glioma là một loại khối u não ác tính phát triển từ các tế bào thần kinh đệm
* meningioma: Là khối u phát triển từ màng não, đa phần là lành tính
* pituitary: Là các khối u xuất hiện ở tuyến yên, nằm ở đáy não.
* space-occupying lesion-: khối bất thường nào chiếm không gian trong não, gây ra sự chèn ép hoặc thay đổi cấu trúc não.
* Train mô hình:

# training yolov5

!python train.py --img 640 --batch 16 --epochs 100  --data data.yaml --weights yolov5s.pt --cache

* Chạy file train.py – là script chính của YOLOv5 dùng để huấn luyện mô hình
* Img 640: kích thước ảnh đầu vào (resize về 640x640 pixel)
* Batch 16: Số lượng ảnh xử lý trong một lần huấn luyện (mini-batch)
* Epochs 100: Huấn luyện mô hình trong 100 vòng lặp
* Chỉ định file data.yaml để mô tả tập dữ liệu
* Weights yolov5.pt: dùng mô hình đã được huấn luyện sẵn trên COCO làm trọng số khởi tạo (pretrained weights).
* Cache: Lưu trữ ảnh và nhãn vào RAM khi bắt đầu huấn luyện, giúp tăng tốc độ trâin (ít phải đọc lại từ ổ đĩa).



* Lưu model đã huấn luyện vào drive

# lưu model đã huấn luyện vào drive

!cp /content/yolov5/runs/train/exp/weights/best.pt /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets

!cp /content/yolov5/runs/train/exp/weights/last.pt /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets

* Lưu model đã huấn luyện sau khi kết thúc quá trình training
* File best.pt là file trọng số tốt nhất, được lưu lại trong quá trình huấn luyện.
* File last.pt là file trọng số cuối cùng sau khi hoàn thành tất cả các epoch (có thể dùng nếu muốn tiếp tục huấn luyện sau này – resume training)
* Chạy thử mô hình đã huấn luyện trên các đối tượng thử nghiệm

# chạy thử

!python detect.py --weights /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets/model/best.pt --img 640 --conf 0.25 --source /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets/images/test

* Chạy script detect.py trong thư mục YOLOv5, dùng để suy luận (inference), tức là chạy mô hình đã train để phát hiện đối tượng.
* Lấy từ file mô hình huấn luyện tốt nhất (best.pt)
* Img 640: resize ảnh đầu vào thành kích thước 640x640 để phù hợp với mô hình (trùng với kích thước khi huấn luyện).
* Conf 0.25 ngưỡng confidence threshold chỉ dự đoán các đối tượng có xác suất >= 25%
* Sử dụng thư mục chứa ảnh dùng để thử nghiệm (tập test).

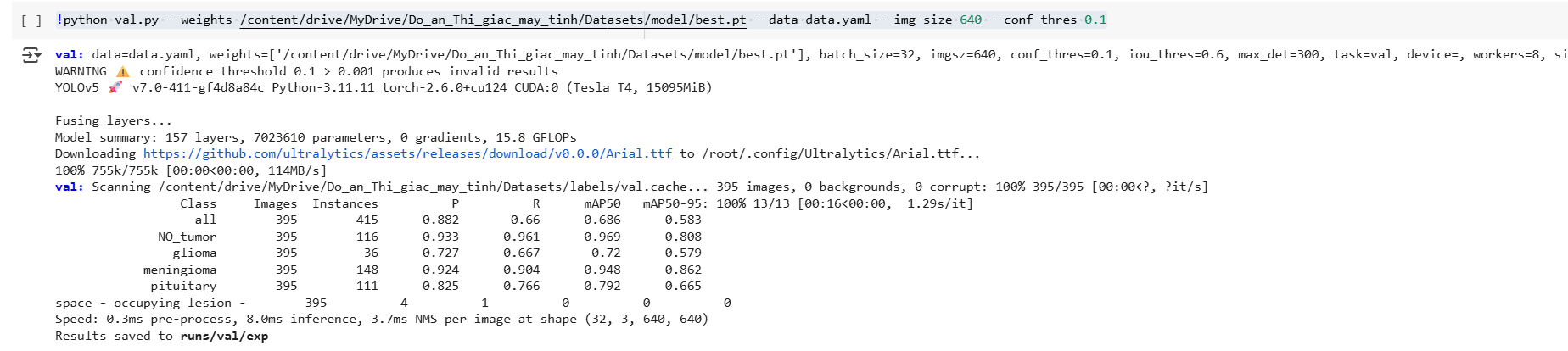
-> Sau khi chạy xong sẽ lưu kết quả vào thư mục runs/detect/exp



1. Đánh giá mô hình

!python val.py --weights /content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets/model/best.pt --data data.yaml --img-size 640 --conf-thres 0.1

* Trong đó:
* Chạy file script val.py trong thư mục YOLOv5: đây là script đánh giá (validation) để tính toán các chỉ số như precision, recall, mAP
* Đánh giá trên mô hình đã huấn luyện tốt nhất (best.pt)
* Conf-thres 0.1: mô hình sẽ xét các dự đoán có độ tin cậy >= 0.1



1. Hiển thị kết quả

# hiện thị kết quả

import cv2

import matplotlib.pyplot as plt

import os

# Đường dẫn tới thư mục chứa kết quả

result\_dir = '/content/yolov5/yolov5/yolov5/runs/detect/exp'  # Hoặc chạy kếtquả nào của bạn

# Lấy danh sách hình ảnh

images = os.listdir(result\_dir)

# Hiển thị từng hình ảnh

for image\_name in images:

    image\_path = os.path.join(result\_dir, image\_name)

    image = cv2.imread(image\_path)

    plt.imshow(cv2.cvtColor(image, cv2.COLOR\_BGR2RGB))

    plt.axis('off')

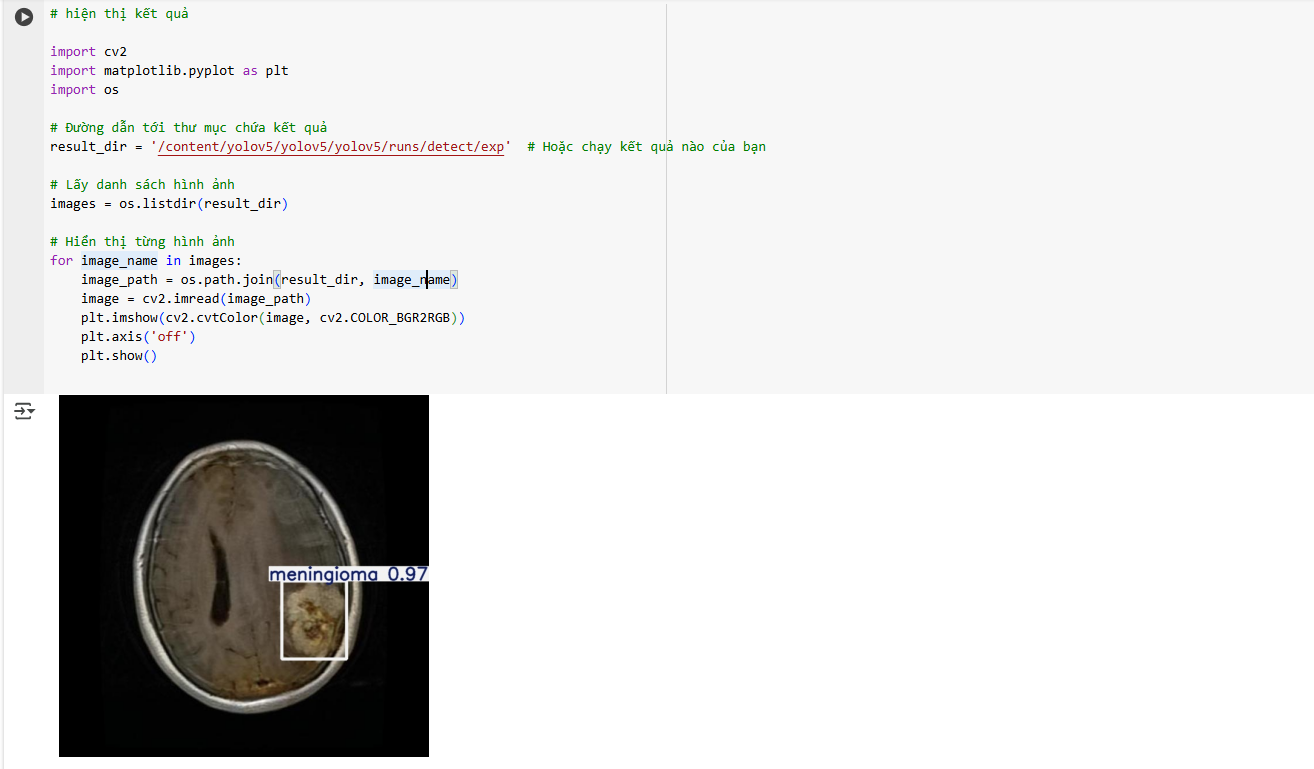
    plt.show()

* Trong đó:
* Khai báo các thư viện cần thiết
* Dẫn đến thư mục chứa kết quả ảnh sau khi chạy detect.py
* Sau đó lấy danh sách tất cả các file trong thư mục kết quả
* Hiển thị ảnh

- Tất cả các ảnh trong file kết quả đều được hiện ra màn hình

- Với hình ảnh cho thấy rõ được khối u đã được bao quanh bởi bording box

- Các hình ảnh được hiển thị kèm theo tên khối u và độ tin cậy của mô hình đối với ảnh đó



1. Xét trên một ảnh hoàn toàn mới

import torch

from PIL import Image

import cv2

import os

from IPython.display import Image as IPyImage, display

# Load model

model = torch.hub.load('ultralytics/yolov5', 'custom', path='/content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/Datasets/model/best.pt', force\_reload=True)

# Chạy detect

results = model('/content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/img5.jpg')

results.save()

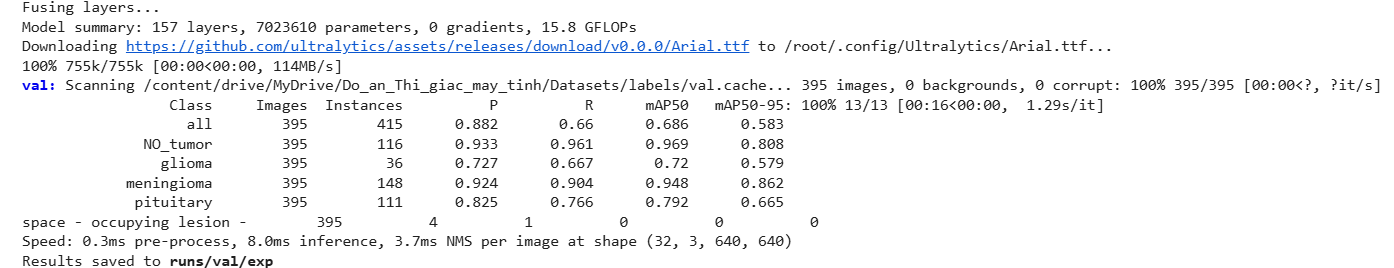
# Hiển thị kết quả

result\_path = 'runs/detect/exp/' + os.path.basename('/content/drive/MyDrive/Do\_an\_Thi\_giac\_may\_tinh/img5.jpg')

display(IPyImage(filename=result\_path))

* Trong đó:
* Khai báo, import thư viện
* Tải mô hình tùy chỉnh với trọng số đã huấn luyện (best.pt)
* Dự đoán trên ảnh img5 có trong thư mục, và lưu ảnh đã gán nhãn vào thư mục runs/detect/exp
* Hiển thị kết quả với kết quả đã được lưu trong thư mục
* Ảnh sẽ được hiện ra với khối u được bao quanh, tên và độ tin cậy.

1. Đánh giá mô hình



1. Tổng quan mô hình

Mô hình đạt kết quả như sau:

* Precision (P): 0.882

Mô hình có độ chính xác cao, tức là khi phát hiện được đối tượng thì phần lớn là đúng.

* Recall (R): 0.660

Tuy nhiên, độ bao phủ lại thấp hơn, mô hình vẫn còn bỏ sót một số đối tượng cần phát hiện.

* [mAP@0.5](mailto:mAP@0.5): 0.686

Mô hình phát hiện tương đối tốt ở mức IoU cơ bản.

* [mAP@0.5-0.95](mailto:mAP@0.5-0.95): 0.583

Mức trung bình khá, phản ánh mô hình khó khăn khi định vị chính xác vị trí bounding box ở các mức IoU cao.

1. Phân tích chi tiết theo từng lớp

* Lớp NO\_tumor
* Precision (P): 0.993
* Recall (R): 0.961
* [mAP@0.5](mailto:mAP@0.5): 0.969
* [mAP@0.5-0.95](mailto:mAP@0.5-0.95): 0.808

=> Mô hình nhận diện rất tốt ở lớp này, mô hình phát hiện chính xác và đầy đủ các đối tượng thuộc lớp này.

* Lớp glioma
* Precision (P): 0.772
* Recall (R): 0.667
* [mAP@0.5](mailto:mAP@0.5): 0.722
* [mAP@0.5-0.95](mailto:mAP@0.5-0.95): 0.579

=> Mô hình nhận diện ở mức trung bình

* Lớp meningioma
* Precision (P): 0.924
* Recall (R): 904
* [mAP@0.5](mailto:mAP@0.5): 0.944
* [mAP@0.5-0.95](mailto:mAP@0.5-0.95): 0.862

=> Hiệu suất rất tốt, ổn định cả về vị trí và phát hiện.

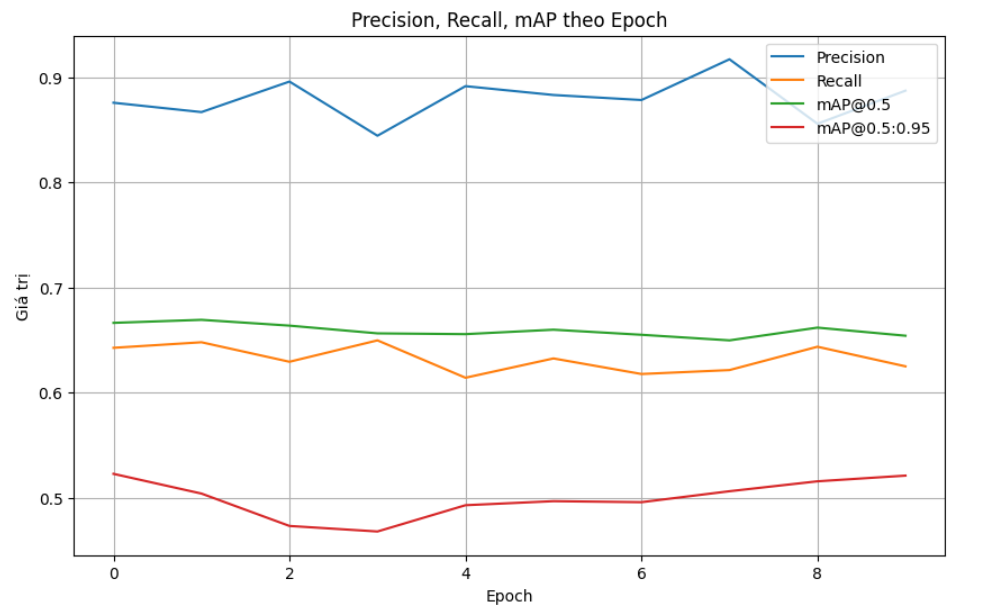
* Lớp pituitary
* Precision (P): 0.825
* Recall (R): 0.766
* [mAP@0.5](mailto:mAP@0.5): 0.792
* [mAP@0.5-0.95](mailto:mAP@0.5-0.95): 0.665

=> Hiệu suất khá, nhưng có thể cải thiện thêm về độ bao phủ (R).

* Lớp space-occupying lesion
* Precision (P): 1.0
* Recall (R): 0.0
* [mAP@0.5](mailto:mAP@0.5): 0.0
* [mAP@0.5-0.95](mailto:mAP@0.5-0.95): 0.0

=> Mặc dù độ chính xác (P) đạt tuyệt đối 1, nhưng độ bao phủ Recall lại bằng 0 cho thấy mô hình không phát hiện được các đối tượng thực tế thuộc lớp này.

1. Đồ thị P, R, [mAP@0.5](mailto:mAP@0.5), [mAP@0.5-0.95](mailto:mAP@0.5-0.95)



Phân tích đồ thị mô hình khi được huấn luyện tiếp tục với 10 epoch

* Precision (Xanh dương):
* Thể hiện tỷ lệ dự đoán đúng trong số các dự đoán dương tính.
* Giá trị cao (~0.87–0.92) → Mô hình ít phát hiện sai (false positive thấp).
* Có sự dao động nhẹ, nhưng nhìn chung là ổn định và cao → tốt.
* Recall (Cam):
* Thể hiện khả năng tìm đúng các đối tượng thực sự có mặt trong ảnh.
* Giá trị ~0.62–0.65 → Mô hình bỏ sót một số đối tượng thật (false negative).
* Đường này dao động nhiều hơn → có thể cần thêm dữ liệu huấn luyện hoặc tăng số epoch.
* mAP@0.5 (Xanh lá):
* Đánh giá tổng thể độ chính xác phát hiện tại mức ngưỡng IoU = 0.5.
* ~0.65–0.67 → Mô hình có khả năng phát hiện mức khá, nhưng chưa thật sự xuất sắc.
* Đường khá ổn định nhưng ít tăng theo thời gian → có thể mô hình đã gần bão hòa hoặc cần tinh chỉnh thêm.
* mAP@0.5:0.95 (Đỏ):
* Chỉ số đánh giá toàn diện nhất, đo ở nhiều ngưỡng IoU khác nhau (0.5 đến 0.95).
* ~0.46–0.53 → Mức trung bình, cho thấy mô hình còn khó khăn trong việc dự đoán chính xác hộp giới hạn (bounding box).
* Có xu hướng tăng dần về sau, tức là mô hình đang học dần các chi tiết phức tạp hơn.

1. Kết luận:

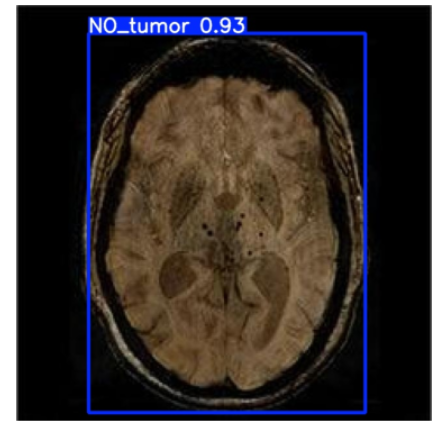
Dựa trên các kết quả đánh giá:

* Mô hình đã học tốt với các lớp có dữ liệu đủ (NO\_tumor, meningioma)
* Các lớp glioma, pituitary có hiệu suất khá nhưng cần cải thiện.
* Mô hình gặp khó khăn với mẫu space-occupying lesion, hầu như bị mất hoàn toàn khả năng phát hiện do dữ liệu huấn luyện quá ít.
* Đề xuất cải thiện:
* Bổ sung dữ liệu, đặc biệt với lớp “space-occupying lession”
* Tăng cường dữ liệu (augmentation) để giúp mô hình học đặc trưng tốt hơn.

1. Hiển thị kết quả đầu ra

Sau khi huấn luyện mô hình và chạy thử, các file được dự đoán với kết quả đầu ra:

* NO\_tumor:



* Nhãn dự đoán: NO\_tumor, mô hình dự đoán ảnh này không có khối u não
* Độ tin cậy (confidence score): 0.93 xác suất mô hình cho rằng ảnh này đúng lớp thuộc lớp NO\_tumor là 93%, tức là rất cao.
* Pituitary:



* Nhãn dự đoán: pituitary, mô hình xác định đây là ảnh chứa khối u tuyến yên
* Độ tin cậy: 0.96 mô hình rất chắc chắn với độ tin cậy lên tới 96%, rất chính xác.
* Menigioma:

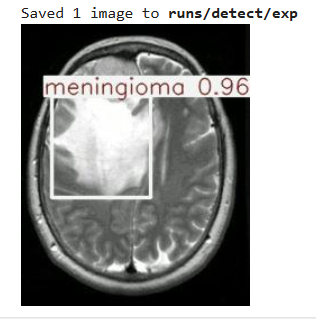


* Nhãn dự đoán: menigioma, mô hình dự đoán đây là một khối u lành tính xuất phát từ màng bao quanh não.
* Với độ tin cậy lên tới 95%, mô hình rất chắc chắn về dự đoán.

=> Mô hình đang dự đoán với độ tin cậy rất cao -> mô hình hoạt động tốt.

1. Hiển thị kết quả với dữ liệu hoàn toàn mới

Khi đưa một ảnh hoàn toàn mới vào mô hình đã huấn luyện thì kết quả nhận được:



* Mô hình dự đoán đây là khối u màng não meningioma
* Với độ tin cậy của mô hình là 96%
* Một sự chắc chắn rất cao về dự đoán này.

1. Kết luận
2. Tổng quan bài toán

Bài toán hướng đến phát hiện và phân loại khối u não qua các lớp như NO\_tumor, glioma, meningioma, space-occupying lession và pituitary. Hiệu suất của mô hình được đánh giá qua các chỉ số:

* Precision (P): Độ chính xác khi dự đoán đúng
* Recall (R): tỷ lệ phát hiện đúng trên tổng số đối tượng thực tế
* mAP@50: Trung bình độ chính xác tại ngưỡng IoU=50%
* mAP@50-95: Trung bình độ chính xác qua nhiều ngưỡng IoU

1. Kết quả

Mô hình có khả năng:

* Phân loại chính xác các loại khối u
* Dự đoán với độ tin cậy cao (confidence từ 85%-99%)
* Hiển thị trực quan kết quả bằng khung bao và nhãn tên u

1. Hạn chế

* Mô hình có thể bị nhiễu do ảnh MRI mờ, chất lượng thấp
* Dữ liệu ở một số lớp còn ít
* Mô hình không hoạt động tốt ở một số lớp đối tượng.