

TRƯỜNG ĐẠI HỌC VĂN LANG
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO ĐỒ ÁN MÔN HỌC HK241
CÁC HỆ HỖ TRỢ RA QUYẾT ĐỊNH (71ITAI40303)

Đề tài:

**XÂY DỰNG HỆ THỐNG TỰ ĐỘNG CHẨN
ĐOÁN BỆNH U NÃO TRONG LĨNH VỰC Y
KHOA DỰA TRÊN MÔ HÌNH CNN**

Nhóm sinh viên thực hiện:

- 1. Đỗ Lý Anh Kiệt - 2274802010451**
 - 2. Nguyễn Công Huy - 2274802010310**
 - 3. Nguyễn Huỳnh Tú - 2174802010332**
- GVHD: Th.s Nguyễn Thái Anh**

TP. Hồ Chí Minh – năm 2024

MỤC LỤC

MỤC LỤC	2
LỜI CẢM ƠN	4
CHƯƠNG 1: MỞ ĐẦU	5
1.1 Lý do chọn chủ đề nghiên cứu	5
1.2 Đối tượng, phạm vi và phương pháp nghiên cứu	5
1.2.1 Đối tượng nghiên cứu	5
1.2.2 Phạm vi nghiên cứu	5
1.2.3 Phương pháp nghiên cứu	6
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	8
2.1 Tổng quan về u não	8
2.1.1 Phương pháp chẩn đoán u não thông qua ảnh chụp MRI trong y tế...8	
2.2 Mô hình Convolutional Neural Networks (CNN)	8
2.2.1 Định nghĩa về Convolutional Neural Networks (CNN)	8
2.2.2 Cấu trúc của Convolutional Neural Networks (CNN)	8
2.3 Xây dựng kiến trúc mô hình với Xception	9
2.4 Lưu mô hình học từ keras	10
2.5 Phát triển hệ thống thành ứng dụng web từ mô hình	11
2.5.1 Flask là gì ?	11
2.5.2 Tại sao áp dụng Flask cho mô hình chẩn đoán u não ?	11
CHƯƠNG 3: PHƯƠNG PHÁP NGHIÊN CỨU	12
3.1 Tiền xử lý dữ liệu	12
3.1.1 Dữ liệu ảnh não chụp MRI từ Kaggle	12
3.1.2 Phân chia dữ liệu	12
3.1.3 Tăng cường dữ liệu	13

3.1.4 Chuẩn hóa dữ liệu.....	14
3.2 Mô hình mạng nơ-ron tích chập cho phân loại bệnh u não.....	14
3.2.1 Định nghĩa các lớp mô hình CNN chẩn đoán u não.....	14
3.2.2 Huấn luyện mô hình	16
3.2.3 Kết quả và đánh giá	16
CHƯƠNG 4: KẾT QUẢ VÀ KẾT LUẬN.....	17
4.1 Kết quả huấn luyện từ mô hình học	17
4.1.1 Đường cong học tập - Learning Curve	17
4.1.2 Ma trận nhầm lẫn - Confusion Matrix.....	19
4.1.3 Báo cáo phân loại	20
4.1.4 Đánh giá mô hình học.....	21
4.1.5 Lưu mô hình học	21
4.2 Triển khai mô hình với hệ thống.....	22
4.2.1 Mô hình hệ thống.....	22
4.2.2 Giao diện trang thông tin	22
4.2.3 Giao diện trang người dùng	23
4.2.4 Thao tác với người dùng.....	24
4.2.5 Kết quả chuẩn đoán từ hệ thống.....	24
4.3 Tóm tắt kết quả thực hiện đồ án.....	25
4.3.1 Kết quả chuẩn đoán từ hệ thống.....	26
TRỌNG SỐ ĐÁNH GIÁ.....	27
TÀI LIỆU THAM KHẢO.....	28

LỜI CẢM ƠN

Viết một báo cáo đồ án môn học là một trong những việc khó nhất mà chúng em phải hoàn thành trong quá trình học một môn học. Trong quá trình thực hiện đề tài chúng em đã gặp rất nhiều khó khăn và bế ngõ. Nếu không có những sự giúp đỡ và lời động viên chân thành của những người bạn học, thầy cô thì có lẽ em khó có thể hoàn thành tốt tiểu luận này. Đầu tiên chúng em xin gửi lời biết ơn chân thành đến thầy Nguyễn Thái Anh, người trực tiếp hướng dẫn chúng em hoàn thành tiểu luận này.

Những ý kiến đóng góp của thầy là vô cùng hữu ích, đã giúp chúng em nhận ra các khuyết điểm của đồ án. Cảm ơn thầy và các bạn trường Đại học Văn Lang là những người đã hỗ trợ chúng em những thắc mắc và trải nghiệm để hoàn thành đồ án môn học.

Nhóm sinh viên thực hiện báo cáo

CHƯƠNG 1: MỞ ĐẦU

1.1 Lý do chọn chủ đề nghiên cứu

Kỹ thuật học sâu hay còn gọi là deep learning, đã tạo ra những bước tiến lớn trong việc giải quyết các bài toán phân tích hình ảnh, đặc biệt trong y tế. Các hệ thống chẩn đoán bệnh dựa trên hình ảnh y tế, như MRI não, đã có thể hỗ trợ bác sĩ đưa ra quyết định tốt hơn về sức khỏe cho bệnh nhân. Các phương pháp này đã cho thấy tỷ lệ thành công cao trong chẩn đoán sớm u não – một căn bệnh nguy hiểm đến tính mạng, nhưng khi phát hiện sớm, cơ hội điều trị thành công sẽ giúp rất nhiều về sau này.

1.2 Đối tượng, phạm vi và phương pháp nghiên cứu

1.2.1 Đối tượng nghiên cứu

Tập dữ liệu Kaggle:

- Kaggle là một nền tảng cộng đồng khoa học dữ liệu, nơi có thể tìm kiếm, chia sẻ và cạnh tranh các tập dữ liệu và mô hình. Các tập dữ liệu trên Kaggle bao gồm nhiều lĩnh vực khác nhau như thị giác máy tính, xử lý ngôn ngữ tự nhiên, dự đoán và phân tích dữ liệu [1].
- Nghiên cứu sẽ tập trung vào việc khám phá, hiểu và sử dụng các tập dữ liệu Kaggle phù hợp cho bài toán phân loại trong học sâu.

Mạng nơ-ron tích chập (Convolutional Neural Networks - CNN):

- CNN là một kiến trúc mạng nơ-ron sâu được sử dụng rộng rãi trong các ứng dụng thị giác máy tính, đặc biệt là phân loại ảnh. CNN sử dụng các lớp tích chập để học các đặc trưng không gian từ dữ liệu ảnh, giúp cải thiện hiệu suất so với các mô hình học sâu truyền thống.
- Việc huấn luyện và tối ưu hóa các mô hình CNN sử dụng các tập dữ liệu từ Kaggle và TensorFlow sẽ giúp thực hiện các tác vụ phân loại ảnh.

Cấu trúc được dùng để xây dựng hệ thống (Flask):

- Flask là một bộ khung nhỏ hay còn được gọi là micro-framework được viết bằng ngôn ngữ lập trình Python, phục vụ cho phát triển web. Nghiên cứu sẽ xem xét môi trường lập trình độc lập của Flask, giúp giảm thiểu lỗi và việc xử lý lỗi dễ dàng hơn. Cuối cùng, đánh giá những ứng dụng thực tiễn sử dụng Flask trong phát triển ứng dụng web [2].

1.2.2 Phạm vi nghiên cứu

Mạng nơ-ron tích chập (Convolutional Neural Networks - CNN):

- Phạm vi nghiên cứu tập trung vào việc áp dụng mô hình CNN để phân tích hình ảnh MRI não, nhằm phát hiện kịp thời loại bệnh và chẩn đoán u não.

Tập dữ liệu từ Kaggle:

- Dữ liệu sẽ là hình ảnh từ Kaggle gồm: tập dữ liệu huấn luyện gọi là Training với 1311 hình ảnh và tập dữ liệu kiểm thử gọi là Testing với 5712 hình ảnh đã được gán bốn loại nhãn tương ứng với bốn loại bệnh u não khác nhau.

1.2.3 Phương pháp nghiên cứu

Nghiên cứu lý thuyết:

Tìm hiểu và nghiên cứu tài liệu về mạng nơ-ron tích chập (Convolutional Neural Networks - CNN) và các kiến trúc CNN phổ biến, như Xception hoặc ResNet. Các kiến thức lý thuyết về học sâu, xử lý ảnh y tế và các kỹ thuật tiền xử lý dữ liệu cũng như tham khảo thêm các tài liệu khoa học và các nguồn học trực tuyến đáng tin cậy như TensorFlow và các mã nguồn từ Kaggle.

Đọc hiểu và tổng hợp các nghiên cứu về chẩn đoán u não dựa trên ảnh MRI, nhằm xác định những ưu, nhược điểm của các phương pháp hiện có và xây dựng nền tảng kiến thức để cải thiện độ chính xác cho mô hình.

Thực hành triển khai:

Tiền xử lý dữ liệu: Tiến hành các bước tiền xử lý cần thiết như thay đổi kích thước ảnh, chuẩn hóa dữ liệu, và tăng cường dữ liệu gọi là data augmentation. Từ đó, cải thiện khả năng khái quát hóa của mô hình.

Xây dựng mô hình CNN: Sử dụng thư viện TensorFlow và Keras để xây dựng mô hình CNN và lựa chọn kiến trúc mô hình phù hợp. Tùy chỉnh các lớp tích chập, lớp gộp (pooling), lớp kết nối đầy đủ (fully connected), và lớp đầu ra để đáp ứng yêu cầu của bài toán phân loại ảnh MRI.

Huấn luyện mô hình: Chia dữ liệu thành các tập huấn luyện, xác thực, và kiểm thử để huấn luyện và đánh giá mô hình. Sử dụng các hàm tối ưu hóa như SGD hoặc Adam và thử nghiệm với các siêu tham số như tốc độ học (learning rate), số lượng mẫu dữ liệu (batch size), số lần duyệt (epoch).

Thử nghiệm và đánh giá:

Đánh giá trên tập kiểm tra: Sau khi huấn luyện, sử dụng tập dữ liệu kiểm tra để đánh giá độ chính xác của mô hình. Tính toán các chỉ số hiệu suất từ tỉ lệ chính xác (accuracy), các giá trị đo (precision), tỉ lệ số điểm (recall), và số dung hòa (F1-score) để có cái nhìn toàn diện về mô hình.

Phân tích kết quả: Sử dụng ma trận nhầm lẫn hay còn gọi là confusion matrix và các biểu đồ mất mát/chính xác để phân tích chi tiết hiệu suất của mô hình, nhận diện các loại u não dễ nhầm lẫn và tìm ra các cải tiến khả thi.

Phát triển hệ thống website: Kết hợp Flask để triển khai mô hình dưới dạng ứng dụng web, tạo giao diện thân thiện giúp người dùng có thể tải ảnh MRI lên và nhận kết quả chẩn đoán.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1 Tổng quan về u não

Định nghĩa: U não là một tập hợp số lượng lớn các tế bào não phát triển bất thường vượt ngoài tầm kiểm soát của cơ thể. Các u não có thể bắt đầu trực tiếp từ tế bào não, tế bào đệm của hệ thần kinh trung ương, hoặc cũng có thể bắt đầu từ các bộ phận khác (ví dụ như thận...) rồi theo máu đến não, được gọi là u di căn não [3].

Các khối u não có thể được phân thành hai loại chính:

- **U lành tính:** là loại u não phát triển chậm, không di căn, có thể được điều trị dứt điểm bằng cách phẫu thuật loại bỏ khối u trực tiếp mà không cần xạ trị hay hóa trị [3].
- **U ác tính:** là loại u não chứa tế bào ung thư (tế bào phân chia nhanh vượt mức kiểm soát). Khối u này dễ tấn công và di căn sang các vùng tế bào khỏe mạnh lân cận. Bệnh phát triển nhanh, dễ tái phát và ảnh hưởng trực tiếp đến tính mạng bệnh nhân [3].

Nguyên nhân và phân loại: Nguyên nhân chính xác gây ra u não vẫn chưa được hiểu rõ, nhưng một số yếu tố rủi ro được cho là có thể góp phần bao gồm di truyền, tiếp xúc với bức xạ và các yếu tố môi trường. U não được phân loại theo vị trí, loại tế bào nguồn gốc và mức độ ác tính.

2.1.1 Phương pháp chẩn đoán u não thông qua ảnh chụp MRI trong y tế

U não là một bệnh lý phức tạp và nguy hiểm, yêu cầu quá trình chẩn đoán và điều trị kịp thời. Chụp cộng hưởng từ (Magnetic Resonance Imaging - MRI) là một phương pháp chính để phát hiện và đánh giá các khối u não. Chẩn đoán bằng MRI cho phép các bác sĩ có thể quan sát được các cấu trúc trong não bộ mà không cần can thiệp phẫu thuật, giúp phát hiện sự tồn tại và đặc điểm của các khối u não. Các hình ảnh MRI cung cấp chi tiết về kích thước, hình dạng và vị trí của u, giúp bác sĩ xác định loại u và đưa ra phương án điều trị phù hợp [4].

2.2 Mô hình Convolutional Neural Networks (CNN)

2.2.1 Định nghĩa về Convolutional Neural Networks (CNN)

CNN được viết tắt của Convolutional Neural Network hay còn được gọi là Mạng nơ-ron tích chập là một trong những mô hình Deep Learning cực kỳ tiên tiến cho phép xây dựng những hệ thống có độ chính xác cao và thông minh. Nhờ khả năng đó, CNN có rất nhiều ứng dụng, đặc biệt là những bài toán cần nhận dạng vật thể (object) trong dữ liệu hình ảnh. Sử dụng các lớp tích chập để tự động và thích nghi học các đặc trưng từ dữ liệu đầu vào, cho phép chúng phát hiện các mẫu phức tạp trong hình ảnh [5].

2.2.2 Cấu trúc của Convolutional Neural Networks (CNN)

Cấu trúc của một mô hình CNN thường bao gồm các lớp chính sau:

- **Convolutional Layers:** là lớp tổng quát đóng vai trò mấu chốt, khi lớp này đảm nhiệm việc thực hiện mọi tính toán như stride, padding, filter map, feature map. Cơ chế của CNN là tạo ra các filter áp dụng vào từng vùng hình ảnh. Các filter map này được gọi là ma trận 3 chiều, bên trong chứa các parameter dưới dạng những con số [6].
- **ReLu layers:** là lớp chỉnh lưu có tác dụng mô phỏng các nơ-ron có tỷ lệ truyền xung qua axon. Trong activation function chúng còn có hàm nghĩa là: Relu, Tanh, Sigmoid, Maxout, Leaky,...
- **Pooling Layers:** là lớp gộp khi nhận phải đầu vào quá lớn, các lớp pooling layer sẽ được xếp giữa những lớp Convolutional layers nhằm mục đích giảm parameter [6].
- **Fully Connected Layers:** là lớp kết nối đầy đủ khi 2 lớp convolutional layer và pooling layer nhận được ảnh truyền, lớp này sẽ có nhiệm vụ xuất kết quả. Khi nhận được kết quả là mô hình đọc được thông tin ảnh, cần phải tạo sự liên kết để cho ra nhiều output hơn. Nếu fully connected layer có dữ liệu về hình ảnh thì chúng sẽ chuyển thành mục chưa được phân chia chất lượng [6].
- **Output Layer:** Lớp này thường sử dụng hàm softmax để dự đoán xác suất của các lớp đầu ra.

2.3 Xây dựng kiến trúc mô hình với Xception

Xception (viết tắt của “Extreme Inception”) là một mô hình mạng nơ-ron sâu được François Chollet giới thiệu, cải tiến từ mô hình Inception của Google. Kiến trúc của Xception kết hợp giữa kỹ thuật tách biệt không gian và kênh màu (depthwise separable convolution) để giảm số lượng tham số và tăng hiệu quả tính toán, đồng thời đạt độ chính xác cao hơn [7].

Kiến trúc Xception sẽ có ba phần chính [10]:

Entry Flow:

- Đầu vào của mạng là ảnh có kích thước cố định (ví dụ: 299x299x3).
- Các tầng đầu tiên sử dụng các convolution tiêu chuẩn để trích xuất đặc trưng ban đầu từ ảnh đầu vào.
- Tiếp theo, mạng sử dụng các convolution tách biệt không gian và kênh màu (depthwise separable convolution) để giảm số lượng tham số mà vẫn giữ được thông tin đặc trưng của ảnh.
- Phần này kết thúc bằng một lớp pooling để giảm kích thước dữ liệu đầu ra, giúp giảm độ phức tạp của các phép tính.

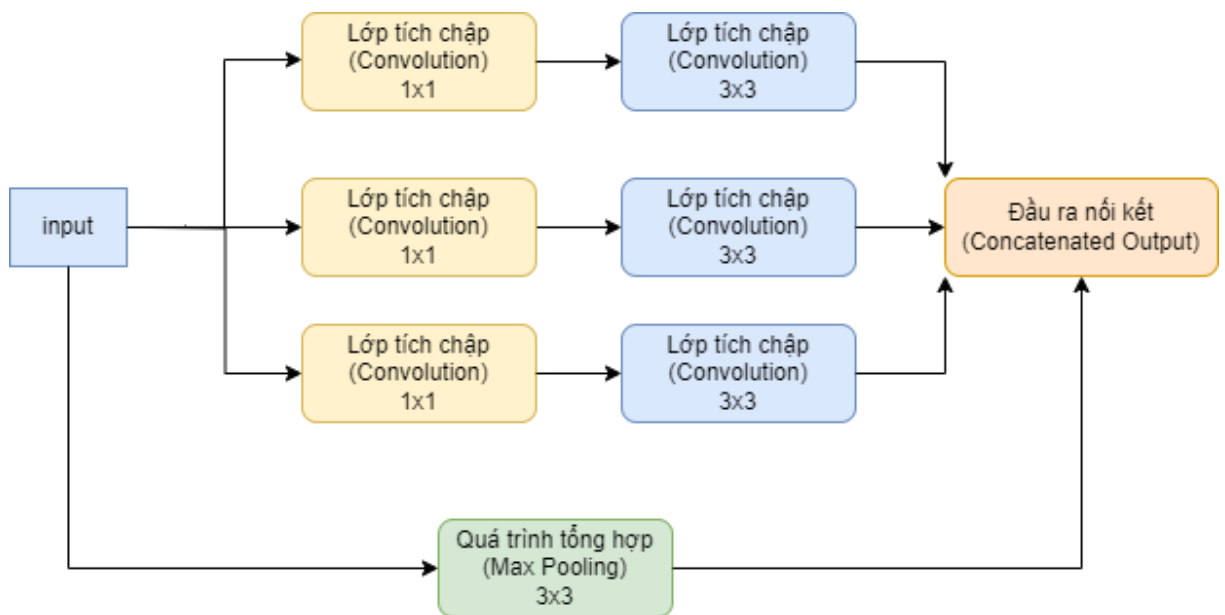
Middle Flow:

- Phần này bao gồm một loạt các tầng convolution tách biệt không gian và kênh màu, được lặp lại nhiều lần.
- Các convolution ở đây được thiết kế để học các đặc trưng sâu hơn của ảnh, giúp tăng cường khả năng phân loại của mô hình.
- Phần Middle Flow có thể được lặp lại nhiều lần tùy theo yêu cầu của ứng dụng, tạo độ sâu cho mạng.

Exit Flow:

- Đây là phần cuối của mạng, nơi các đặc trưng sâu từ phần Middle Flow được xử lý để chuẩn bị cho phần phân loại.
- Các lớp convolution tách biệt không gian và kênh màu được áp dụng thêm một lần nữa, sau đó được theo sau bởi một lớp pooling.
- Cuối cùng, các đặc trưng này được chuyển đổi thành một vector đầu ra phẳng thông qua Global Average Pooling.
- Lớp cuối cùng là một lớp dense (hoặc fully connected) với số đầu ra tương ứng với số lượng lớp cần phân loại.

Sơ đồ cơ bản về kiến trúc Xception



2.4 Lưu mô hình học từ keras

Sau khi huấn luyện mô hình, triển khai lên hệ thống bằng cách lưu mô hình dưới dạng đuôi tệp là .h5 vì đây là định dạng của thư viện tensorflow đã sử dụng.

H5 là một trong những Định dạng dữ liệu phân cấp (HDF) được sử dụng để lưu trữ lượng lớn dữ liệu. Nó được sử dụng để lưu trữ một lượng lớn dữ liệu dưới dạng mảng nhiều chiều. Định dạng này chủ yếu được sử dụng để lưu trữ dữ liệu khoa học được tổ chức tốt để truy xuất và phân tích nhanh chóng. H5 được giới

thiệu là định dạng tệp nâng cao hơn cho H4. Ban đầu nó được phát triển bởi Trung tâm ứng dụng siêu máy tính quốc gia và hiện được hỗ trợ bởi Tập đoàn HDF [8].

2.5 Phát triển hệ thống thành ứng dụng web từ mô hình

2.5.1 Flask là gì ?

Flask là một micro-framework được viết bằng ngôn ngữ lập trình Python dùng cho các nhà phát triển web. Được phát triển bởi Armin Ronacher. Flask dựa trên bộ công cụ Werkzeug WSGI và template engine Jinja2. Micro ở đây không có nghĩa là framework này thiếu các chức năng mà thể hiện ở việc nó sẽ cung cấp những chức năng “ cốt lõi ” nhất cho các ứng dụng web và có khả năng mở rộng, người dùng cũng có thể mở rộng bất cứ lúc nào vì Flask hỗ trợ rất nhiều các tiện ích mở rộng như tích hợp CSDL, hệ thống upload, xác thực, template, email... Việc là một micro-framework cũng giúp cho flask có một môi trường xử lý độc lập và ít phải sử dụng các thư viện bên ngoài, điều này giúp nó nhẹ và ít gặp các lỗi hơn, việc phát hiện và xử lý các lỗi cũng dễ dàng và đơn giản hơn [2].



2.5.2 Tại sao áp dụng Flask cho mô hình chẩn đoán u não ?

Flask có thể tích hợp mô hình .h5 của CNN nhờ vào khả năng hỗ trợ các thư viện như Keras và TensorFlow, cho phép tải và sử dụng mô hình dễ dàng. Thêm vào đó, Flask cho phép xây dựng API RESTful, giúp nhận dữ liệu đầu vào từ người dùng và trả về kết quả dự đoán. Việc triển khai ứng dụng Flask cũng rất đơn giản, có thể thực hiện nhanh chóng trên máy chủ hoặc dịch vụ đám mây. Tóm lại, chỉ cần tải mô hình vào Flask và tạo endpoint là có thể tích hợp thành công mô hình học sâu vào ứng dụng web.

CHƯƠNG 3: PHƯƠNG PHÁP NGHIÊN CỨU

3.1 Tiền xử lí dữ liệu

3.1.1 Dữ liệu ảnh não chụp MRI từ Kaggle

Trong nghiên cứu này, sử dụng bộ dữ liệu **Brain Tumor MRI Dataset** với 7022 ảnh MRI u não từ Kaggle các hình ảnh thuộc loại không có khối u được lấy từ tập dữ liệu Br35H [9].

Bộ dữ liệu **Brain Tumor MRI Dataset** có tổng cộng **7022 hình ảnh MRI** với đuôi tệp là jpg, được phân loại thành bốn nhóm và được gán bốn nhãn như sau:

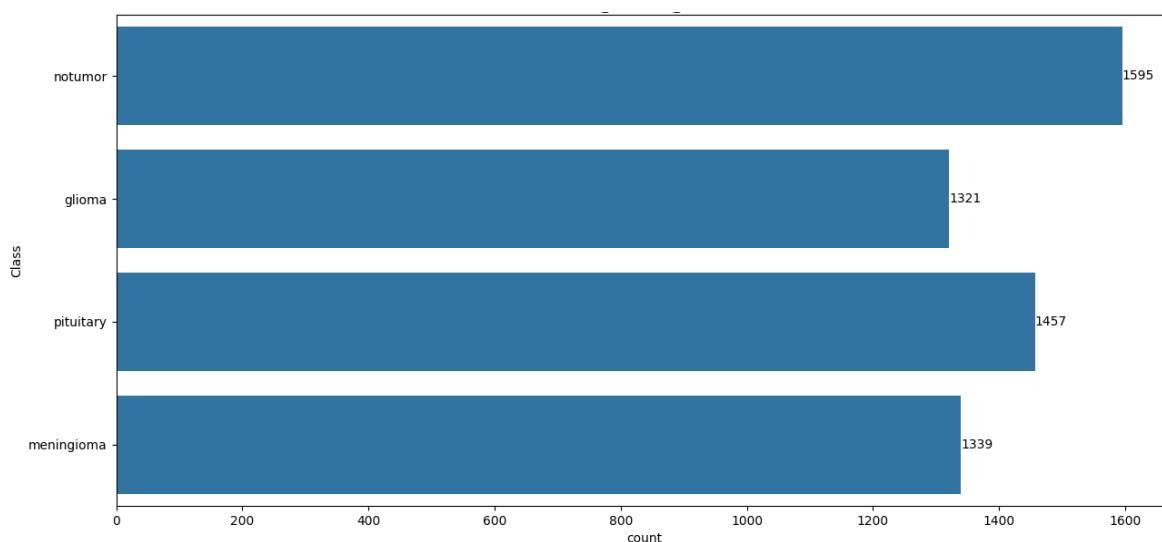
- Glioma: là bệnh u phát triển từ các tế bào đệm của não.
- Meningioma: là một loại u xuất phát từ màng não.
- Pituitary: u tuyến yên phát triển từ tuyến yên.
- No tumor: Các trường hợp không có sự xuất hiện của khối u trong não.

3.1.2 Phân chia dữ liệu

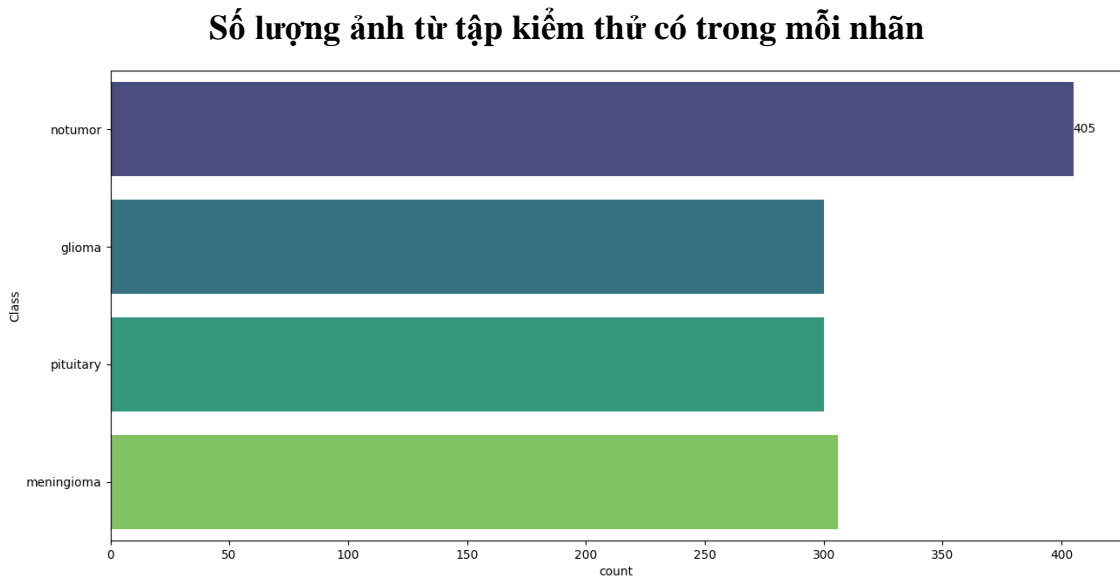
Tập dữ liệu được chia thành các tập huấn luyện, đánh giá và kiểm thử để đảm bảo tính khách quan và độ chính xác của mô hình. Cụ thể:

- Dữ liệu từ tập huấn luyện hay còn gọi là training: sẽ có 5712 ảnh cho bốn nhãn u não.

Số lượng ảnh từ tập huấn luyện có trong mỗi nhãn



- Dữ liệu từ tập huấn luyện hay còn gọi là testing: được sử dụng để điều chỉnh các siêu tham số và đánh giá mô hình trong quá trình huấn luyện gồm 1311 ảnh.



Hai tập này sẽ được chia với xác suất là 8/2.

3.1.3 Tăng cường dữ liệu

Với bước đầu tiên của xử lý dữ liệu sẽ tách tập kiểm thử 1311 ảnh với `train_size` là 0.5 thành tập dữ liệu đánh giá. Ở đây `train_size` có nghĩa là là **số xác định kích thước của tập huấn luyện** [10]. Từ đó tập kiểm thử sẽ được chia nhỏ ra thành tập đánh giá và kiểm thử.

Sử dụng seed gọi là hạt giống ngẫu nhiên với số lượng là 50 cùng với biến `random_state` là trạng thái ngẫu nhiên là `random_state=seed`. Lí do sử dụng `random_state` cùng với seed sẽ tối ưu dữ liệu học cho mô hình khi huấn luyện vì các tập huấn luyện và tập kiểm tra sẽ được gán nhãn giả khác nhau qua các lần thực thi khác nhau và quá trình xáo trộn sẽ được kiểm soát để mô hình không lấy cái phân nhãn giả bị trùng [11].

Tiếp theo định nghĩa **hàm tăng cường dữ liệu** dựa trên thư viện `tensorflow.keras` hàm sẽ có các lớp cấu trúc như sau:

- `RandomFlip`: lật hình theo chiều ngang và chiều dọc.
- `RandomRotation`: xoay hình với một góc ngẫu nhiên (tối đa 20%).
- `RandomZoom`: phóng to hình với một tỉ lệ ngẫu nhiên (tối đa 10%).
- `RandomContrast`: tăng hoặc giảm độ tương phản của hình ảnh (tối đa 20%).
- `Random Translation`: lấy mẫu ngẫu nhiên để dịch một hình ảnh trong không gian hai chiều. Với hai tham số 0.1, 0.1 biểu thị mức độ dịch chuyển theo trục X và trục Y.
- Phần đa dạng về tăng cường có thể thêm vào cho mô hình tiếp theo là phép biến đổi đàn hồi trên các hình ảnh đầu vào cho hàm `elastic_transform`, nơi `image` là hình ảnh đang xử lý. Với `tf.float32` là kiểu dữ liệu của đầu ra.

Điều này có nghĩa là đầu ra của hàm `elastic_transform` sẽ được coi là một tensor có kiểu `float32`.

Ta có hàm lớp nhiễu: `noise_factor = 0.15`. Hằng số xác định độ lớn của nhiễu sẽ được thêm vào dữ liệu. Giá trị `0.15` có nghĩa là nhiễu sẽ được điều chỉnh với tỷ lệ 15%.

Bước cuối khi đưa dữ liệu vào mô hình huấn luyện ta sẽ khởi tạo số lượng hình ảnh sẽ được xử lý cùng một lúc trong mỗi bước huấn luyện. Ở đây, giá trị là 32 có nghĩa `batch_size = 32`

Thay đổi kích thước ảnh cho phù hợp với mô hình học bằng cách đưa giá trị điểm về khoảng từ 0 đến 1 bằng cách chia cho 255 là giá trị tối đa của điểm ảnh.

3.1.4 Chuẩn hóa dữ liệu

Các tham số của hàm chuẩn hóa dữ liệu được định nghĩa như sau:

```
_gen = ImageDataGenerator(  
    rescale=1/255,  
    brightness_range=(0.8, 1.2),  
    rotation_range=15,  
    width_shift_range=0.15,  
    height_shift_range=0.15,  
    shear_range=0.15,  
    zoom_range=0.15,  
    horizontal_flip=True,  
    fill_mode="nearest")  
  
valid_ts_gen = ImageDataGenerator(rescale=1/255)
```

rescale=1/255: là bước chuẩn hóa dữ liệu bằng cách chia giá trị của mỗi pixel trong hình ảnh cho 255, chuyển đổi phạm vi giá trị từ `[0, 255]` về `[0, 1]`. giúp giảm sự biến động giữa các giá trị của điểm ảnh và cải thiện hiệu quả khi huấn luyện mô hình.

3.2 Mô hình mạng nơ-ron tích chập cho phân loại bệnh u não

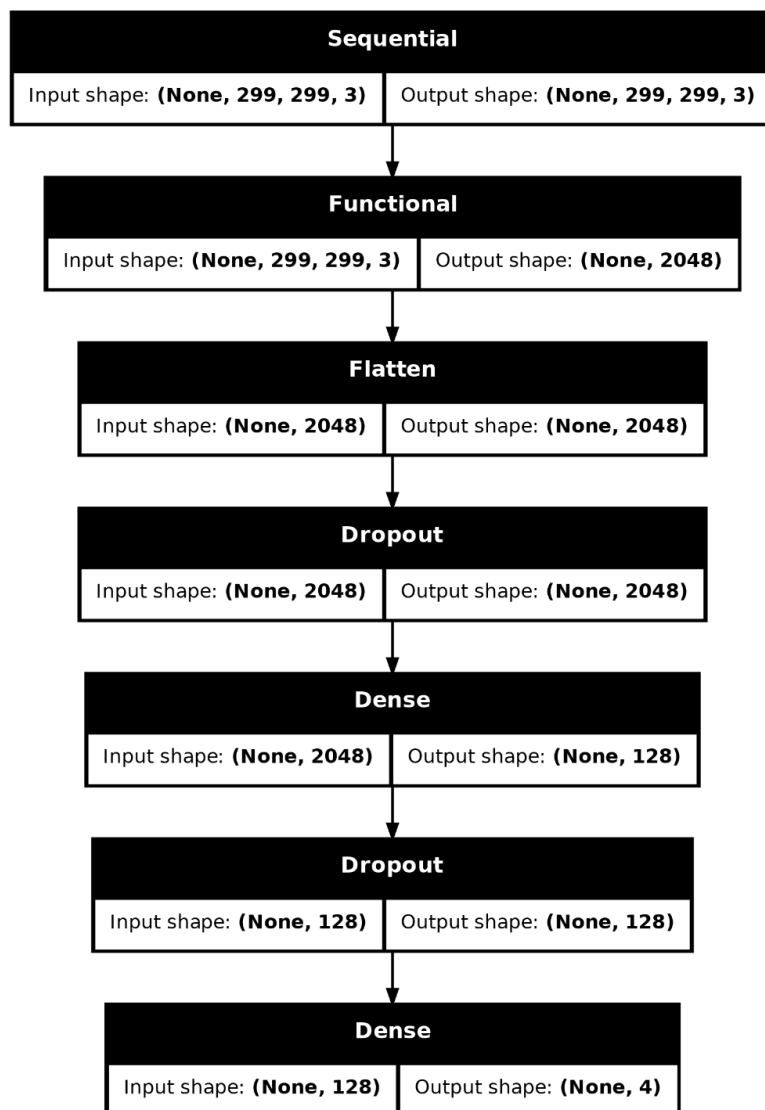
3.2.1 Định nghĩa các lớp mô hình CNN chẩn đoán u não

Với lớp cơ bản của mô hình là `base_model` đã có. Từ đây, các tham số mô hình mạng nơ-ron tích chập được định nghĩa như sau:

- Flatten: là lớp làm phẳng này chuyển đổi một tensor nhiều chiều là kết quả đầu ra từ `base_model` thành một vector 1 chiều.

- Dropout: là lớp bỏ qua sử dụng để giảm thiểu hiện tượng quá khớp trong quá trình huấn luyện mô hình. Nó sẽ bỏ qua 30% tại $\text{rate} = 0.3$ số lượng nơ-ron trong lớp này một cách ngẫu nhiên trong mỗi bước huấn luyện.
- Dense(128, activation='relu'): đây là một lớp nơ-ron kết nối đầy đủ với 128 nơ-ron. Hàm kích hoạt **relu** (Rectified Linear Unit) nghĩa là Đơn vị tuyến tính chỉnh lưu được sử dụng để thêm tính phi tuyến tính vào mô hình, giúp nó học các đặc trưng phức tạp hơn.
- Dropout(rate= 0.25): tương tự như lớp Dropout trước đó, lớp này sẽ bỏ qua 25% số nơ-ron trong lớp này trong mỗi bước huấn luyện, nhằm mục đích giảm thiểu quá khớp.
- Dense(4, activation='softmax'): lớp dày đặc là lớp đầu ra của mô hình với 4 nơ-ron, ứng với 4 lớp phân loại nhãn đầu ra. Thêm hàm kích hoạt là softmax sẽ biến đổi đầu ra thành xác suất cho mỗi lớp, cho phép mô hình có thể phân loại dữ liệu vào một trong bốn nhãn.

Tổng quan về lớp mô hình



3.2.2 Huấn luyện mô hình

Ước lượng động lượng thích ứng hay còn gọi là **Adaptive Moment Estimation** gọi tắt là hàm tối ưu **Adam** là một phương pháp tối ưu hóa ngẫu nhiên, một quá trình quan trọng trong học sâu và học máy. Tính chất đệ quy này rất phù hợp để giải quyết các hệ phương trình tuyến tính có dữ liệu nhiều và xấp xỉ các giá trị cực trị của các hàm chỉ có thể được ước lượng qua các quan sát có nhiễu [12].

Ta sẽ biên dịch thuật toán tối ưu như sau:

- Thuật toán tối ưu Adam sẽ có tốc độ học gọi là learning_rate (lr) sẽ có tham số là 0.001. Sử dụng hàm mất mát **Categorical Cross-Entropy loss** từ thư viện tensorflow hay còn được gọi là Softmax loss. Đây là hàm loss thông dụng nhất được sử dụng cho bài toán phân loại đa lớp. Bao gồm hàm kích hoạt softmax tác động vào output của lớp kết nối đầy đủ cuối cùng của mạng là logit và cross-entropy loss với công thức tính như sau:

$$L_i = -\frac{1}{N} \sum_{i=1}^N \log \frac{e^{W_{y_i}^T x_i + b_{y_i}}}{e^{\sum_{j=1}^n W_j^T x_i + b_j}}$$

Dựa vào thuật toán trên quy trình sẽ được cải thiện đáng kể về tốc độ học của mô hình được huấn luyện trên tập dữ liệu.

Bước cuối cùng là thêm dừng huấn luyện sớm nếu mô hình không cải thiện sau một số epoch nhất định. Epoch là một lần duyệt qua hết các dữ liệu trong tập huấn luyện thông qua theo dõi giá trị mất mát trên tập xác thực, đặt patience=5 nếu không có cải thiện nào sau 5 epoch, quá trình huấn luyện sẽ dừng lại và trả về trọng số tốt nhất của mô hình khi dừng. Thêm **ReduceLROnPlateau** Giảm tốc độ học khi mô hình không cải thiện. Đặt **factor = 0.5** cho tốc độ học sẽ bị giảm một nửa nếu không có cải thiện. và khai báo **patience = 3** nếu không có cải thiện trong 3 epoch, tốc độ học sẽ được giảm. **min_lr = 1e-6**: Tốc độ học tối thiểu sẽ không thấp hơn giá trị này.

Đánh giá: hiệu suất của mô hình sẽ được đánh giá trên tập kiểm tra để đo lường hiệu suất thực tế. Thông qua các biểu đồ học và ma trận nhầm lẫn được tạo ra để phân tích chi tiết kết quả. Các chỉ số như độ chính xác, độ nhạy, độ đặc hiệu, và số dung hòa được sử dụng để đánh giá tổng quan về hiệu suất của mô hình.

3.2.3 Kết quả và đánh giá

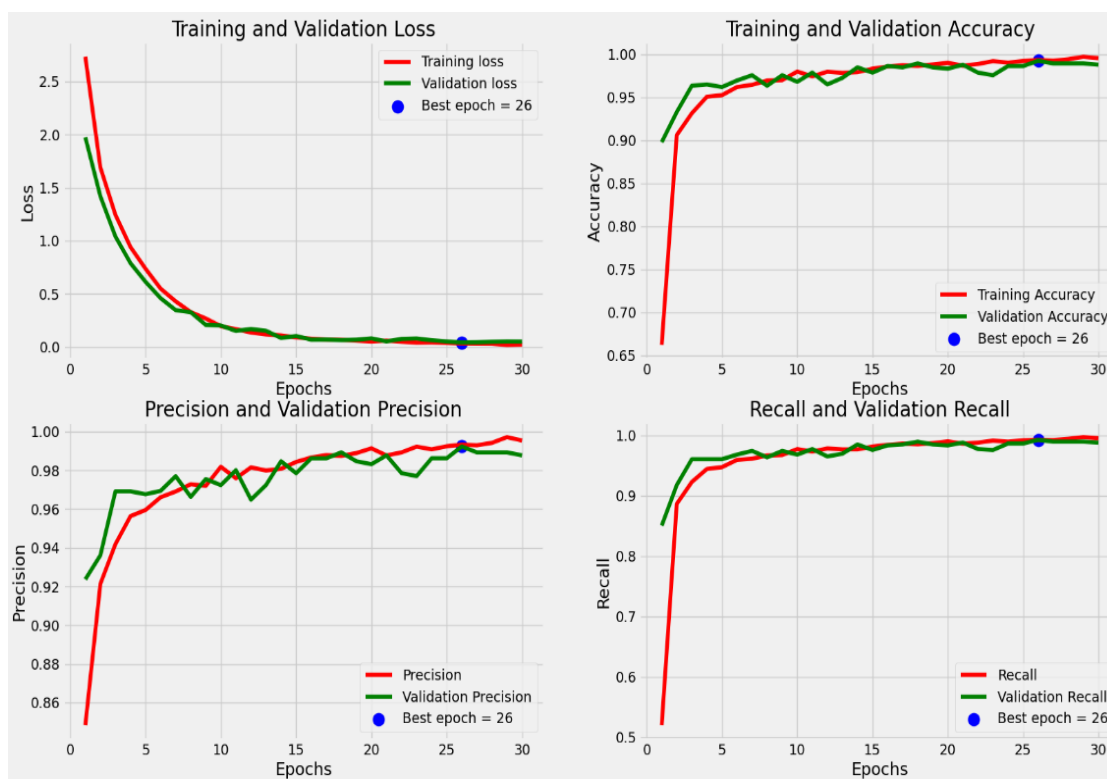
Mô hình đạt độ chính xác cao trong phân loại các loại u não, tuy nhiên vẫn cần cải thiện để đánh giá tốt hơn các biến thể của bệnh. Kết quả này cho thấy tiềm năng ứng dụng trong quản lý và điều trị u não. **Lưu mô hình học** là bước khởi đầu quan trọng để triển khai hệ thống trong thực tế.

CHƯƠNG 4: KẾT QUẢ VÀ KẾT LUẬN

4.1 Kết quả huấn luyện từ mô hình học

4.1.1 Đường cong học tập - Learning Curve

Biểu đồ đường cong học tập của mô hình qua từng lần lặp



Với 30 lần lặp thì mô hình sẽ được dừng ở mức lặp cao nhất là 26

Training and Validation Loss (Biểu đồ đầu tiên, bên trái trên cùng):

Đường màu đỏ (Training Loss): Cho biết mức độ lỗi của mô hình trên tập dữ liệu huấn luyện ở mỗi lần lặp.

Đường màu xanh lá (Validation Loss): Cho biết mức độ lỗi của mô hình trên tập dữ liệu kiểm thử (validation) ở mỗi epoch.

Ý nghĩa: Giúp theo dõi quá trình tối ưu hóa của mô hình. Lúc đầu, cả training loss và validation loss đều giảm khi mô hình học. Nếu hai đường này hội tụ và giảm đều, điều đó chỉ ra rằng mô hình đang học tốt mà không bị quá khớp (overfitting). Nếu validation loss bắt đầu tăng trong khi training loss tiếp tục giảm, đó là dấu hiệu của hiện tượng overfitting.

Training and Validation Accuracy (Biểu đồ thứ hai, bên phải trên cùng):

Đường màu đỏ (Training Accuracy): Cho thấy độ chính xác của mô hình trên tập dữ liệu huấn luyện ở mỗi epoch.

Đường màu xanh lá (Validation Accuracy): Cho thấy độ chính xác của mô hình trên tập dữ liệu kiểm tra (validation) ở mỗi epoch.

Ý nghĩa: Biểu đồ này biểu thị khả năng của mô hình trong việc phân loại chính xác các mẫu dữ liệu. Nếu cả training và validation accuracy đều tăng và hội tụ về giá trị cao, điều đó chứng tỏ mô hình đang hoạt động tốt. Nếu validation accuracy không tăng hoặc giảm trong khi training accuracy vẫn tăng, đó là dấu hiệu của overfitting.

Precision and Validation Precision (Biểu đồ thứ ba, bên trái dưới cùng):

Đường màu đỏ (Precision): Biểu thị độ chính xác của mô hình trên tập dữ liệu huấn luyện. Precision đo lường tỷ lệ dự đoán đúng trên tổng số các dự đoán dương tính của mô hình.

Đường màu xanh lá (Validation Precision): Biểu thị độ chính xác của mô hình trên tập dữ liệu kiểm tra (validation).

Ý nghĩa: Precision cao cho thấy mô hình ít khi đưa ra dự đoán sai. Nếu đường validation precision bám sát hoặc dao động gần đường training precision, điều đó chứng tỏ mô hình học tốt mà không bị overfitting.

Recall and Validation Recall (Biểu đồ thứ tư, bên phải dưới cùng):

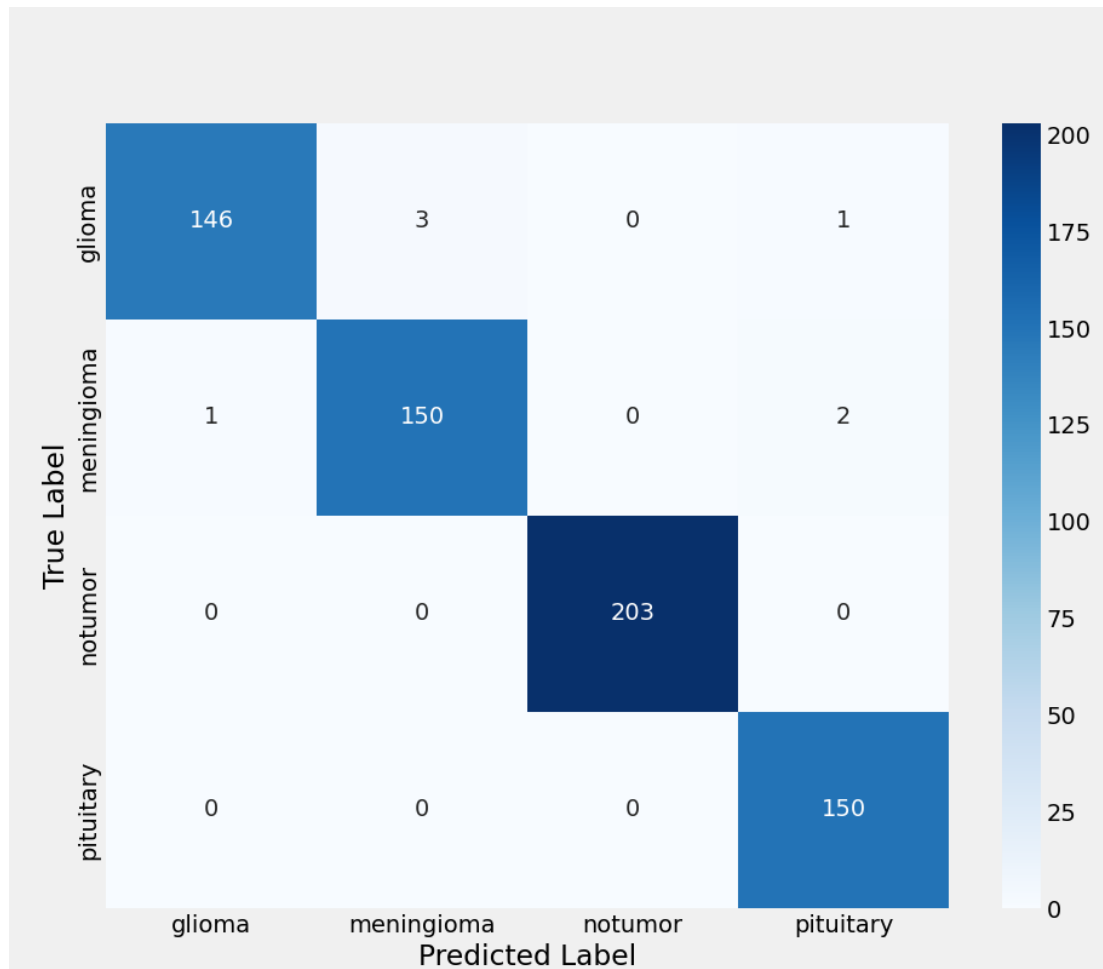
Đường màu đỏ (Recall): Biểu thị khả năng của mô hình trong việc phát hiện tất cả các mẫu dương tính trong tập dữ liệu huấn luyện.

Đường màu xanh lá (Validation Recall): Biểu thị khả năng của mô hình trong việc phát hiện tất cả các mẫu dương tính trong tập dữ liệu kiểm tra (validation).

Ý nghĩa: Recall cao có nghĩa là mô hình có thể phát hiện được nhiều trường hợp dương tính. Nếu đường validation recall dao động gần đường training recall, điều đó chứng tỏ mô hình đang hoạt động tốt mà không gặp vấn đề lớn về overfitting.

Tổng kết: Các biểu đồ này giúp bạn theo dõi quá trình học của mô hình, xác định xem mô hình đang tối ưu hóa tốt hay có dấu hiệu của việc học chưa đủ (underfitting) hoặc quá khớp (overfitting). Dấu chấm xanh ở mỗi biểu đồ thể hiện điểm tốt nhất (epoch tốt nhất) mà mô hình đạt được trên tập kiểm tra (validation).

4.1.2 Ma trận nhầm lẫn - Confusion Matrix



Cấu trúc của ma trận nhầm lẫn:

- **Hàng (True Label):** Đại diện cho nhãn thực tế (nhãn đúng) của dữ liệu kiểm thử.
- **Cột (Predicted Label):** Đại diện cho nhãn mà mô hình đã dự đoán.

Mỗi ô trong ma trận thể hiện số lượng mẫu dữ liệu thực tế hàng và số lượng mẫu dữ liệu dự đoán ở cột. Các ô nằm trên đường chéo chính của ma trận thể hiện các dự đoán đúng, còn các ô ngoài đường chéo thể hiện các dự đoán sai.

- Ở dòng đầu tiên, mô hình đã dự đoán đúng 146 trường hợp là glioma. Có 3 trường hợp glioma bị dự đoán nhầm là meningioma. Có 0 trường hợp bị nhầm thành "notumor". Có 1 trường hợp bị nhầm thành pituitary.
- Tiếp theo là nhãn thứ hai mô hình đã dự đoán đúng 150 trường hợp là meningioma. Có 1 trường hợp meningioma bị nhầm thành glioma. Có 0 trường hợp bị nhầm thành "notumor". Có 2 trường hợp bị nhầm thành pituitary.

- Nhãn thứ ba mô hình đã dự đoán đúng tất cả 203 trường hợp là "notumor". Không có trường hợp nào bị nhầm thành glioma, meningioma, hoặc pituitary.
- Nhãn cuối cùng (pituitary): Mô hình đã dự đoán đúng tất cả 150 trường hợp là pituitary. Không có trường hợp nào bị nhầm thành glioma, meningioma hoặc "notumor".

Tổng quan: Chỉ có một số ít trường hợp bị nhầm lẫn giữa glioma và meningioma hoặc glioma với pituitary. Mô hình không gặp nhiều khó khăn trong việc phân biệt "notumor" và pituitary.

4.1.3 Báo cáo phân loại

	precision	recall	f1-score	support
glioma	0.99	0.97	0.98	150
meningioma	0.98	0.98	0.98	153
notumor	1.00	1.00	1.00	203
pituitary	0.98	1.00	0.99	150
accuracy			0.99	656
macro avg	0.99	0.99	0.99	656
weighted avg	0.99	0.99	0.99	656

Precision (Độ chính xác): là tỉ lệ số điểm true positive trong số những điểm được phân loại là positive (TP + FP).

$$\text{Công thức: Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Precision cao đồng nghĩa với việc độ chính xác của các điểm tìm được là cao.

Recall (Độ nhạy): Recall được định nghĩa là tỉ lệ số điểm true positive trong số những điểm thực sự là positive (TP + FN).

$$\text{Công thức: Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Recall cao đồng nghĩa với việc True Positive Rate cao, tức tỉ lệ bỏ sót các điểm thực sự positive là thấp.

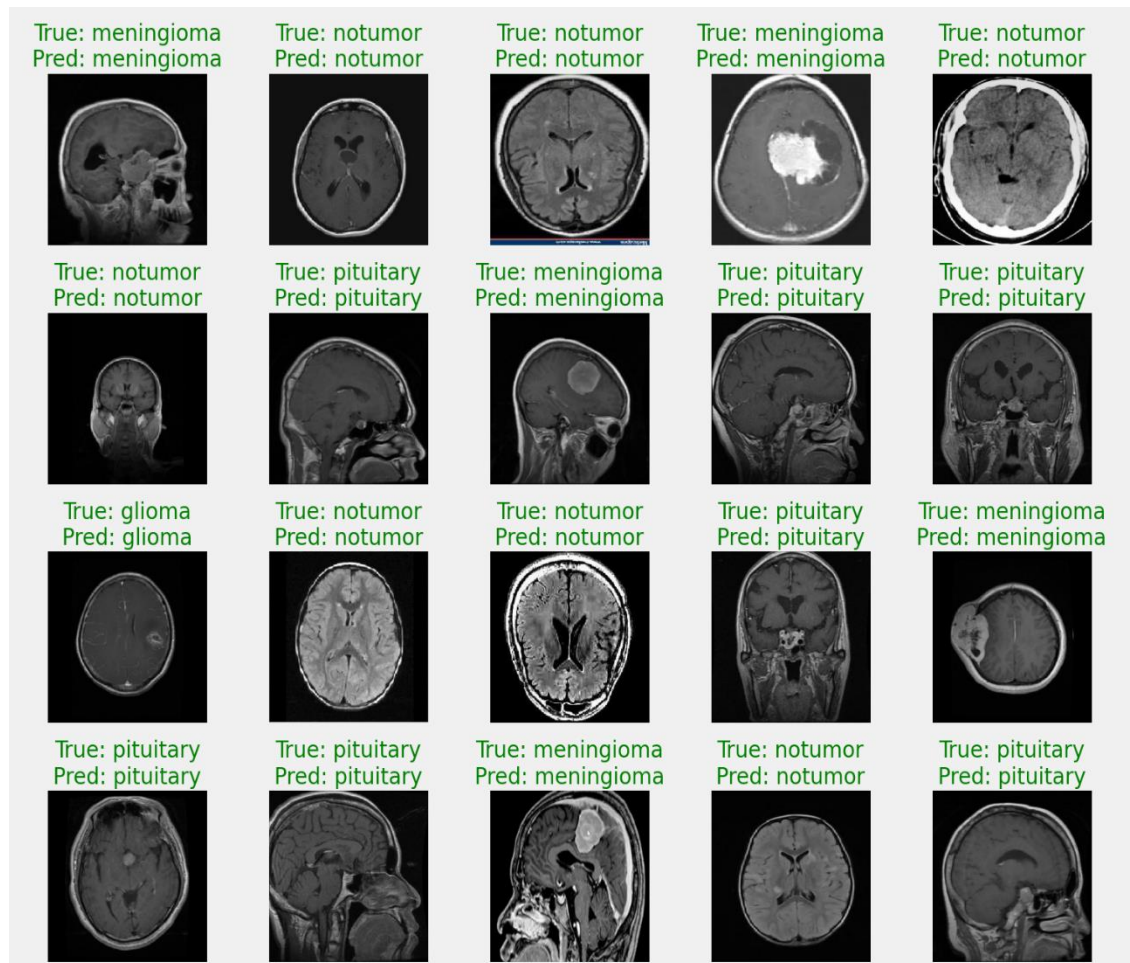
F1-Score: Trung bình điều hòa của Precision và Recall.

$$\text{Công thức: F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Support: Số lượng mẫu thực tế trong từng lớp.

4.1.4 Đánh giá mô hình học

Biểu đồ dự đoán từ các nhãn bệnh



Train Loss rất nhỏ 0.0248 cho thấy mô hình đã học rất tốt trên tập dữ liệu huấn luyện.

Train Accuracy đạt 99.58% nghĩa là mô hình đã phân loại đúng gần như toàn bộ các mẫu trong tập huấn luyện.

Validation Loss thấp 0.0438 và gần với train loss cho thấy mô hình không bị overfitting, tức là mô hình đã tổng quát hóa tốt trên tập xác thực (validation).

Validation Accuracy là 99.24%, chỉ thấp hơn một chút so với train accuracy.

Test Loss: 0.0439

Test Accuracy: 98.93%

4.1.5 Lưu mô hình học

Do sử dụng thư viện tensorflow nên việc lưu mô hình dưới dạng đuôi .h5 là cần thiết. H5 là một trong những Định dạng dữ liệu phân cấp (HDF) được sử dụng để lưu trữ lượng lớn dữ liệu. Nó được sử dụng để lưu trữ một lượng lớn dữ liệu dưới dạng mảng nhiều chiều. Định dạng này chủ yếu được sử dụng để lưu trữ dữ liệu khoa học được tổ chức tốt để truy xuất và phân tích nhanh chóng [8].

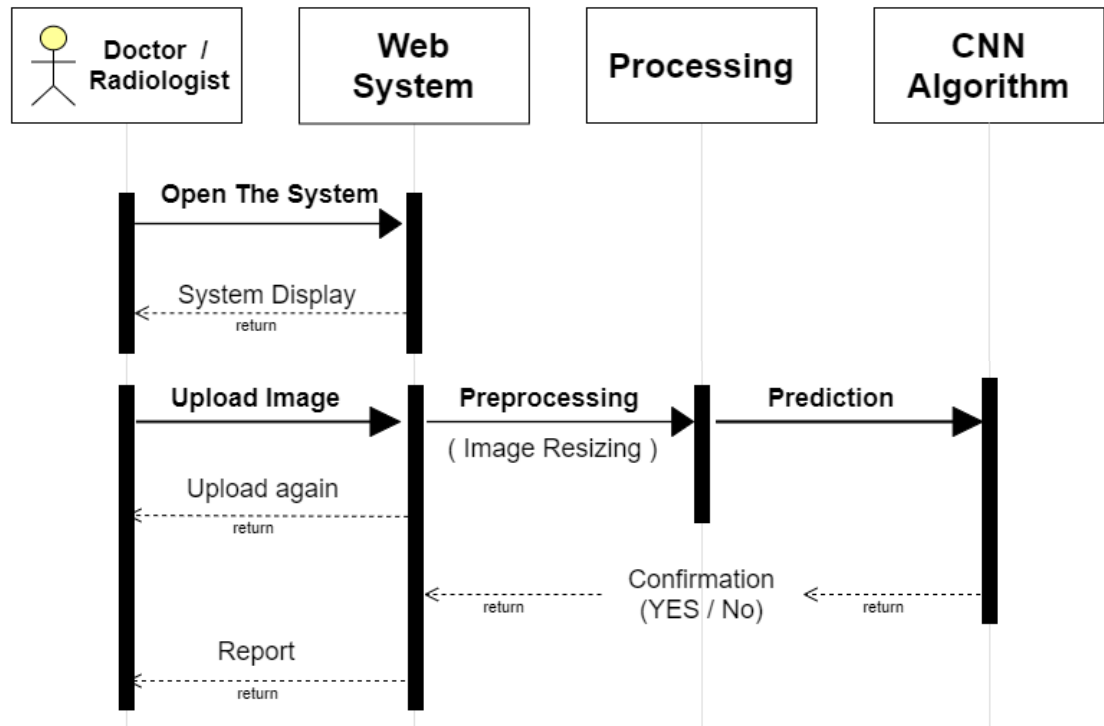
Save the model

```
model.save('brain_tumor_classifier.h5')

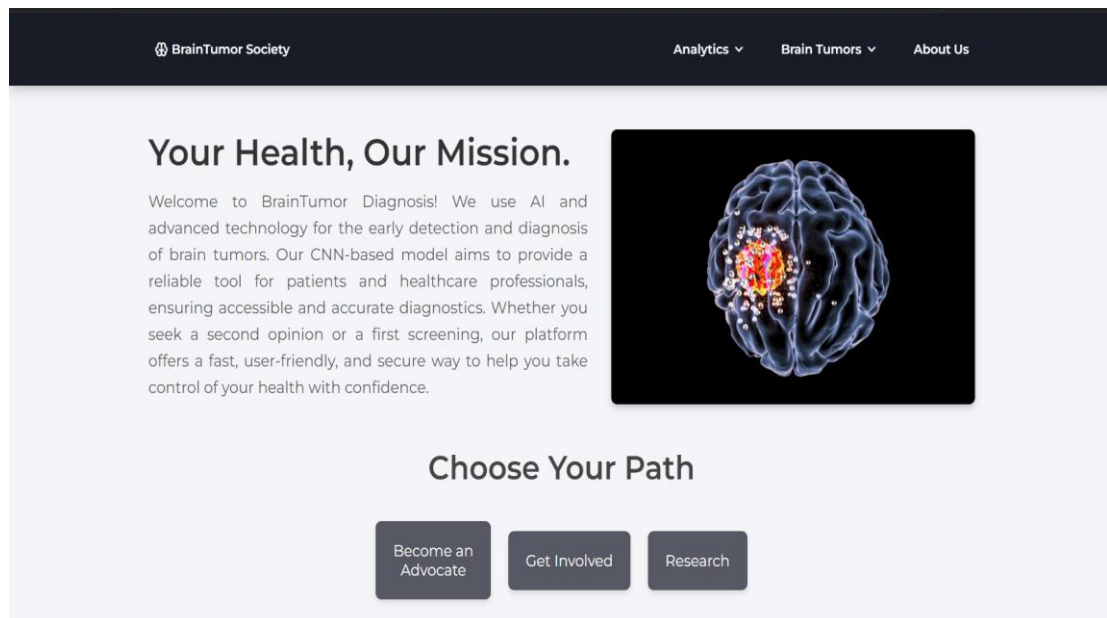
print("Model saved as brain_tumor_classifier.h5")
```

4.2 Triển khai mô hình với hệ thống

4.2.1 Mô hình hệ thống



4.2.2 Giao diện trang thông tin

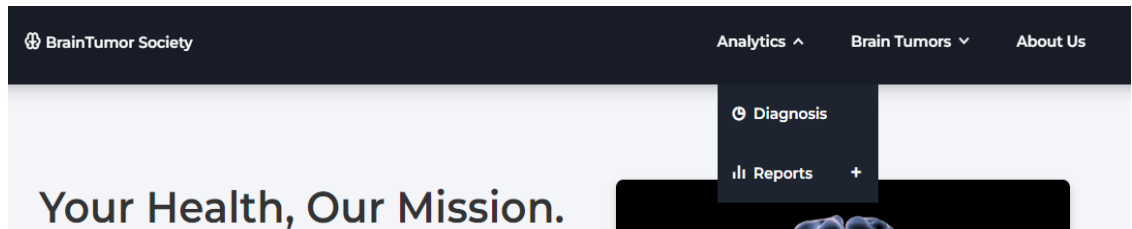


Trang chủ sẽ là một trang khuyến nghị thông tin cho người dùng

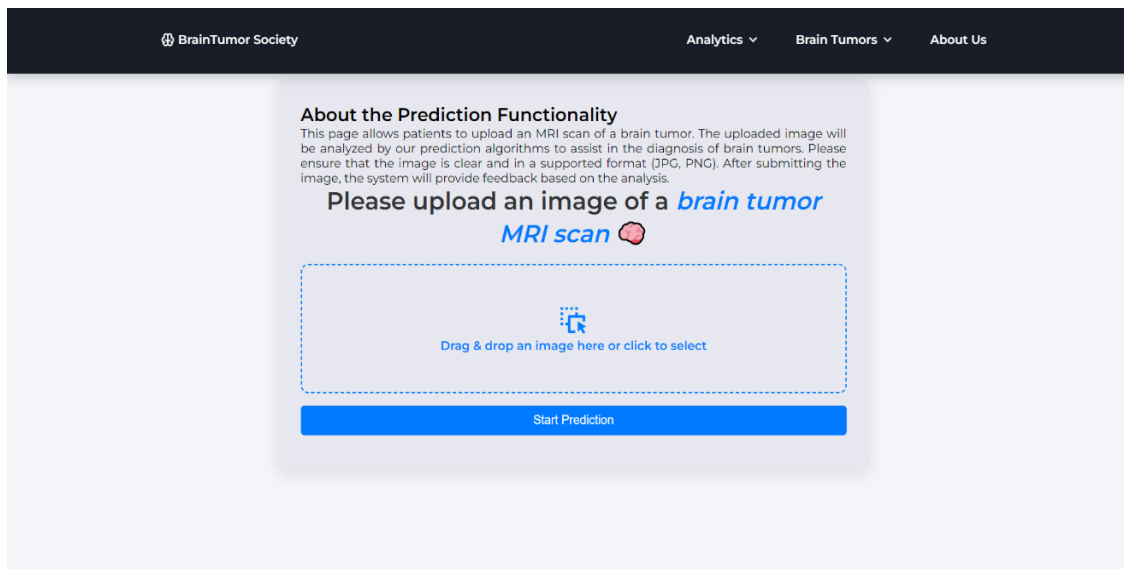
4.2.3 Giao diện trang người dùng

Trang dự đoán của mô hình sẽ nằm ở mục Analytics gọi là mục phân tích. Trong mục chính của Analytics sẽ có một mục Diagnosis là mục chẩn đoán, người dùng sẽ thông tin vào trang Diagnosis để bắt đầu chẩn đoán bệnh.

Thông tin chỉ mục



Trang chẩn đoán



Khi vào trang chẩn đoán người dùng sẽ được hướng dẫn cách tải ảnh và chẩn đoán kết quả từ hình ảnh của mình.

4.2.4 Thao tác với người dùng

About the Prediction Functionality
This page allows patients to upload an MRI scan of a brain tumor. The uploaded image will be analyzed by our prediction algorithms to assist in the diagnosis of brain tumors. Please ensure that the image is clear and in a supported format (JPG, PNG). After submitting the image, the system will provide feedback based on the analysis.

Please upload an image of a *brain tumor* MRI scan 🧠


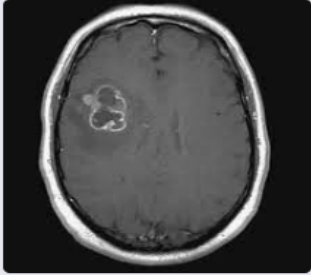

Drag & drop an image here or click to select

Image Preview:



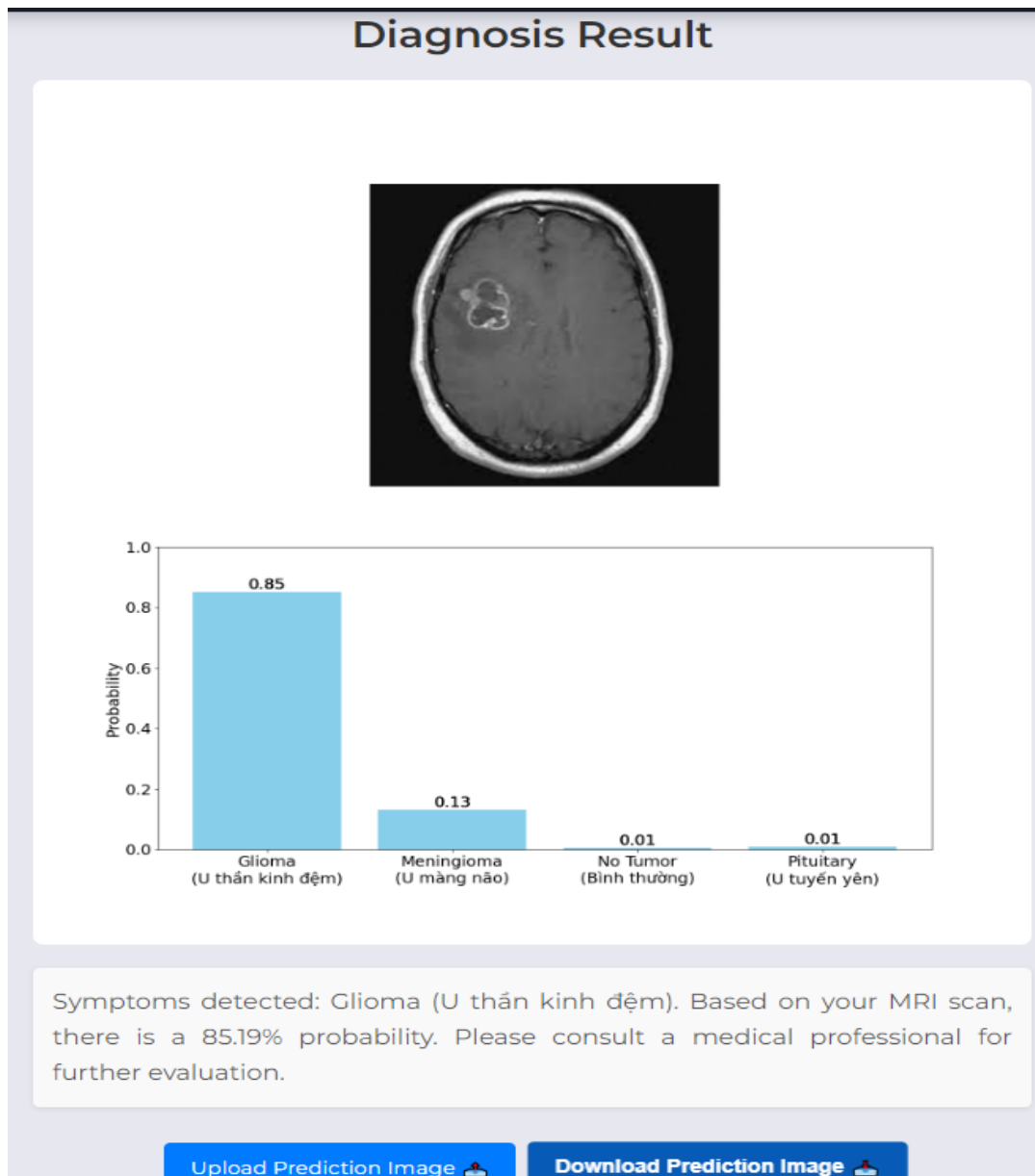
Remove Image

Start Prediction

Các chức năng sẽ được hiển thị đơn giản cho người dùng có thể chỉnh sửa và xem trước thông tin tải lên hoặc xóa và sửa. Qua đó bắt đầu chẩn đoán ở nút xanh dương bắt đầu dự đoán (Start Prediction).

4.2.5 Kết quả chuẩn đoán từ hệ thống

Kết quả sẽ được trả về như ảnh mô tả thông tin bên dưới cùng với các biểu đồ chẩn đoán cho người dùng nhìn tổng quát hơn về loại bệnh mà đang mắc phải cùng với đó là dòng chữ kết luận như sau:



Triệu chứng được phát hiện: U thần kinh đệm (Glioma). Dựa trên kết quả chụp MRI của bạn, có 85,19% xác suất. Vui lòng tham khảo ý kiến của chuyên gia y tế để được đánh giá thêm.

Bên cạnh đó, người dùng cũng có thể tải ảnh báo cáo từ hệ thống để in ra và lưu lại để mang đến cho các chuyên gia đánh giá và chẩn đoán bệnh hiệu quả hơn.

4.3 Tóm tắt kết quả thực hiện đồ án

Dự án xây dựng hệ thống chẩn đoán khối u não dựa trên mô hình mạng nơ-ron tích chập đã triển khai một cách thành công, mang lại nhiều lợi ích cho người dùng và góp phần phát hiện bệnh và điều trị kịp thời cho người dùng.

Dữ liệu hình ảnh của các bãi đỗ xe được thu thập từ các camera giám sát.

Tiền xử lý dữ liệu hình ảnh để phù hợp với yêu cầu của mô hình mạng nơ-ron tích chập, kiểm tra các ảnh MRI và chẩn đoán các khối u não.

Sử dụng các mô hình đã có kiến trúc như Xception để cải thiện độ chính xác tăng cường các lớp cho mô hình học.

Huấn luyện các mô hình với dữ liệu đã chuẩn bị, sử dụng các kỹ thuật tối ưu hóa như EarlyStopping và ReduceLROnPlateau.

Đánh giá độ chính xác và hiệu suất của mô hình trên tập dữ liệu kiểm tra.

Triển khai mô hình lên hệ thống trang web đơn giản giúp hỗ trợ chẩn đoán cho người dùng bằng cách nhận diện và phân loại kết quả từ mô hình mạng nơ-ron tích chập cho chẩn đoán u não.

4.3.1 Kết quả chuẩn đoán từ hệ thống

Tuy nhiên, vẫn còn nhiều khía cạnh cần được phát triển và cải tiến để hoàn thiện hệ thống.

Hạn chế về tài nguyên: với việc gần như mô hình có thể bị khớp dữ liệu thì việc cần thêm dữ liệu trên mười ngàn ảnh là cần thiết nhưng thiết bị như là GPU khi tiếp xúc với số lượng quá lớn dẫn đến mô hình không thể được huấn luyện.

Mở rộng phân tích dữ liệu: dữ liệu vẫn còn thiếu khá nhiều biến thể từ bốn chủng bệnh chính gây ra sự nhầm lẫn cho việc học của mô hình. Cho nên, cần phải có nguồn dữ liệu với độ chính xác cao và phân thêm nhiều nhãn cho mô hình để có thể dễ dàng tiếp cận cho mô hình hơn.

Cải thiện hiệu suất mô hình: Tiếp tục tinh chỉnh và tối ưu hóa các mô hình mạng nơ-ron tích chập để đạt độ chính xác và hiệu suất cao hơn.

Tăng cường giao diện người dùng: Phát triển giao diện người dùng thân thiện và trực quan hơn, cung cấp thông tin chi tiết về khuyến nghị cho người dùng để kịp thời xử lý các tình huống phát hiện u não muộn dẫn đến giai đoạn nặng.

TRỌNG SỐ ĐÁNH GIÁ

Tên	Nội dung công việc	Phần trăm hoàn thành
Đỗ Lý Anh Kiệt	Huấn luyện và đánh giá mô hình, viết báo cáo, xây dựng hệ thống website.	100%
Nguyễn Công Huy	Thu thập và xử lý dữ liệu.	100%
Nguyễn Huỳnh Tú	Tiền xử lý và chuẩn bị dữ liệu, huấn luyện mô hình, xây dựng PowerPoint.	100%

TÀI LIỆU THAM KHẢO

- [1] Coursera, "What Is Kaggle and What Is It Used For?," [Online]. Available: <https://www.coursera.org/articles/kaggle>.
- [2] M. Academy, "Python Flask Là Gì? So Sánh Flask và Django," [Trực tuyến]. Available: <https://www.mcivietnam.com/blog-detail/flask-python-la-gi-so-sanh-flask-va-django/>.
- [3] Bệnh Viện Tâm Anh, "Tâm Anh Hospital," [Trực tuyến]. Available: <https://tamanhhospital.vn/u-nao/>.
- [4] V. I. Hospital, "VinMec International Hospital," [Trực tuyến]. Available: <https://www.vinmec.com/vie/bai-viet/khi-nao-ban-can-chup-cong-huong-tu-mri-vi>.
- [5] VIETNIX, "Định nghĩa CNN là gì?," [Trực tuyến]. Available: <https://vietnix.vn/cnn-la-gi/>.
- [6] VIETNIX, "Các lớp cơ bản của mạng CNN là gì?," [Trực tuyến]. Available: <https://vietnix.vn/cnn-la-gi/#cac-lop-co-ban-cua-mang-cnn-la-gi>.
- [7] M. Nickparvar, "Brain Tumor MRI Dataset," [Online]. Available: <https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset?select=Training>.
- [8] M. Stojiljković, "Split Your Dataset With scikit-learn's train_test_split()," [Online]. Available: <https://realpython.com/train-test-split-python-data/>.
- [9] R. Pramoditha, "Why do we set a random state in machine learning models?," [Online]. Available: <https://towardsdatascience.com/why-do-we-set-a-random-state-in-machine-learning-models-bb2dc68d8431>.
- [10] V. AI, "Paper reading | Xception phiên bản nâng cấp của Inception V3," [Online]. Available: <https://viblo.asia/p/paper-reading-xception-phien-ban-nang-cap-cua-inception-v3-MkNLr1ZoJgA>.
- [11] MarketMuse, "Adaptive Moment Estimation (ADAM)," [Online]. Available: <https://blog.marketmuse.com/glossary/adaptive-moment-estimation-adam-definition/>.