

TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

=====***=====



ĐỒ ÁN TỐT NGHIỆP
NGÀNH: KHOA HỌC MÁY TÍNH

ĐỀ TÀI
NGHIÊN CỨU VÀ PHÁT TRIỂN ỨNG DỤNG NHẬN DẠNG U
NĂO TỪ ẢNH MRI BẰNG MẠNG DENSENET

Giáo viên hướng dẫn: TS. Nguyễn Mạnh Cường

Lớp: KHMT01 – K16

Sinh viên thực hiện: Đào Trọng Hoàn

Mã sinh viên: 2021606290

Hà Nội - 2025

MỤC LỤC

DANH MỤC HÌNH ẢNH.....	4
DANH MỤC NHỮNG TỪ VIẾT TẮT	7
DANH MỤC BẢNG BIỂU.....	8
LỜI NÓI ĐẦU.....	10
CHƯƠNG 1: KHẢO SÁT VÀ PHÁT BIỂU BÀI TOÁN	14
1.1 Khảo sát hiện trạng y tế Việt Nam	14
1.1.1 Hiện trạng y tế Việt Nam.....	14
1.1.2 Ứng dụng công nghệ cao chuẩn đoán bệnh ở việt nam.....	15
1.2 Bài toán nhận dạng ảnh u não.....	16
1.2.1 Khái quát về u não	16
1.2.2 Hiện trạng u não ở Việt Nam.....	21
1.3 Phát Biểu Bài Toán.....	21
1.3.1 Đầu vào	21
1.3.2 Đầu ra.....	22
1.3.3 Các ràng buộc của bài toán.....	23
1.3.4 Mục tiêu của bài toán.....	24
1.3.5 Cơ hội và thách thức.....	25
1.4 Tóm tắt chương.....	26
CHƯƠNG 2: CÁC KỸ THUẬT GIẢI QUYẾT BÀI TOÁN.....	28
2.1 Phương hướng tiếp cận bài toán	28
2.2 Một số kỹ thuật giải quyết bài toán	28
2.3 K-Means Clustering.....	29
2.3.1 Định nghĩa	29
2.3.2 Cách hoạt động	30

2.3.3	Công thức.....	30
2.3.4	Ưu điểm	31
2.3.5	Nhược điểm	31
2.4	Mạng nơ ron tích chập.....	31
2.4.1	Tổng quan	31
2.4.2	Kiến trúc	32
2.5	Mạng nơ-ron ResNet50	44
2.5.1	Giới thiệu chung về ResNet50.....	44
2.5.2	Ý tưởng chính: Residual Learning	44
2.5.3	Kiến trúc tổng thể của ResNet50.....	44
2.5.4	Bottleneck block trong ResNet50.....	45
2.5.5	Ưu điểm của ResNet50.....	46
2.5.6	Nhược điểm của ResNet50.....	47
2.6	Mạng nơron DenseNet 121	47
2.6.1	Người giới thiệu và thời gian.....	47
2.6.2	Ý tưởng chính của DenseNet.....	48
2.6.3	Số lượng tham số	49
2.6.4	Kiến trúc DenseNet-121	49
2.6.5	Ưu điểm của DenseNet.....	51
2.6.6	Nhược điểm của DenseNet	52
2.7	Tóm tắt chương.....	52
CHƯƠNG 3: THỰC NGHIỆM		54
3.1	Môi trường thực nghiệm.....	54
3.2	Dữ liệu thực nghiệm	55
3.2.1	Nội dung	55
3.2.2	Cấu trúc thư mục	56

3.3	Tiền xử lý dữ liệu.....	56
3.4	Huấn luyện với mô hình DenseNet 121	59
3.5	Huấn luyện với mô hình ResNet50	62
3.6	Các kết quả thực nghiệm	64
3.7	Tóm tắt chương.....	70
CHƯƠNG 4: XÂY DỰNG SẢN PHẨM		72
4.1	Flask.....	72
4.1.1	Giới thiệu về Flask.....	72
4.1.2	Các tính năng của Flask Framework:	72
4.2	Bootstrap.....	73
4.2.1	Giới thiệu về Bootstrap.....	73
4.2.2	Mục đích và vai trò	73
4.2.3	Các thành phần chính.....	74
4.3	Thiết kế giao diện hệ thống	74
4.3.1	Giao diện đăng nhập	74
4.3.2	Giao diện dự đoán hình ảnh u não	76
4.4	Kiểm thử	79
4.4.1	Kiểm thử giao diện đăng nhập.....	79
4.4.2	Kiểm thử giao diện dự đoán hình ảnh u não.....	84
4.5	Tóm tắt chương.....	87
KẾT LUẬN		89
TÀI LIỆU THAM KHẢO.....		90

DANH MỤC HÌNH ẢNH

Hình 1.1 Ảnh u não.....	17
Hình 1.2 U thần kinh đệm.....	17
Hình 1.3 U màng não.....	18
Hình 1.4 U tuyến yên.....	19
Hình 2.1 Hình ảnh các khối u với nhãn	28
Hình 2.2 Mô hình K-Means	30
Hình 2.3 Minh họa kiến trúc mạng CNN	33
Hình 2.4 Minh họa nhân tích chập.....	34
Hình 2.5 Hàm kích hoạt ReLU	35
Hình 2.6 Hàm kích hoạt Leaky ReLU	36
Hình 2.7 Hàm kích hoạt Sigmoid	37
Hình 2.8 Hàm kích hoạt Tanh.....	38
Hình 2.9 Minh họa MaxPooling	39
Hình 2.10 Minh họa Average pooling	40
Hình 2.11 Minh họa phép làm phẳng	41
Hình 2.12 Minh họa tầng kết nối đầy đủ	42
Hình 2.13 Mô hình ResNet50	45
Hình 2.14 Mô hình DenseNet-121.....	50
Hình 3.1 Một số hình ảnh trong bộ dữ liệu.....	55
Hình 3.2 Các tham số của ImageDataGenerator	57
Hình 3.3 Tạo Generator từ thư mục ảnh.....	58

Hình 3.4 Định nghĩa thư mục dữ liệu	59
Hình 3.5 Xây dựng mô hình	59
Hình 3.6 Biên dịch mô hình.....	60
Hình 3.7 Thiết lập callbacks	60
Hình 3.8 Huấn luyện mô hình.....	60
Hình 3.9 Quá trình huấn luyện mô hình DenseNet121	61
Hình 3.10 Dự đoán với 10 hình ảnh trong tập test	61
Hình 3.11 Ma trận nhầm lẫn của mô hình	62
Hình 3.12 Cấu trúc mô hình ResNet50.....	63
Hình 3.13 Quá trình huấn luyện mô hình ResNet50	64
Hình 3.14 Biểu đồ Accuracy của DenseNet121	65
Hình 3.15 Biểu đồ Loss của DenseNet121	65
Hình 3.16 Biểu đồ Accuracy của ResNet50	67
Hình 3.17 Biểu đồ Loss của ResNet50.....	68
Hình 3.18 So sánh giữa hai mô hình.....	69
Hình 4.1 Flask	72
Hình 4.2 Bootstrap	73
Hình 4.3 Màn hình đăng nhập	75
Hình 4.4 Màn hình thông báo yêu cầu đăng nhập	76
Hình 4.5 Màn hình phân loại u não	76
Hình 4.6 Màn hình trả về dự đoán khối u.....	78
Hình 4.7 Màn hình dự đoán và chi tiết khối u	78
Hình 4.8 Màn hình đăng nhập	79

Hình 4.9 Màn hình sau khi đăng nhập thành công	80
Hình 4.10 Màn hình khi đăng nhập sai tên người dùng	81
Hình 4.11 Màn hình khi nhập sai mật khẩu	82
Hình 4.12 Màn hình đăng nhập với thông tin trống tên đăng nhập.....	83
Hình 4.13 Màn hình đăng nhập với thông tin trống mật khẩu	83
Hình 4.14 Màn hình dự đoán thành công	85
Hình 4.15 Màn hình dự đoán nhưng không chọn ảnh	86
Hình 4.16 Màn hình hiển thị kết quả dự đoán	87

DANH MỤC NHỮNG TỪ VIẾT TẮT

CNN	Convolutional Neural Networks
K-means	K-number of means
ReLU	Rectified Linear Unit
GPU	Graphics Processing Unit

DANH MỤC BẢNG BIỂU

Bảng 2.1 Bảng mô tả kiến trúc mạng ResNet50.....	45
Bảng 2.2 Bảng mô tả các thành phần ở mô hình DenseNet-121	50
Bảng 3.1 Bảng thông tin bộ dữ liệu	55

LỜI CẢM ƠN

Lời đầu tiên cho phép em gửi lời cảm ơn sâu sắc tới các thầy cô trong Trường Công nghệ thông tin và truyền thông - Trường Đại học Công Nghiệp Hà Nội, những người đã hết mình truyền đạt và chỉ dẫn cho chúng em những kiến thức, những bài học quý báu và bổ ích. Đặc biệt em xin được bày tỏ sự tri ân và xin chân thành cảm ơn giảng viên T.S Nguyễn Mạnh Cường người trực tiếp hướng dẫn, chỉ bảo em trong suốt quá trình học tập, nghiên cứu và hoàn thành được đồ án. Sau nữa, em xin gửi tình cảm sâu sắc tới gia đình và bạn bè vì đã luôn bên cạnh khuyến khích, động viên, giúp đỡ cả về vật chất lẫn tinh thần cho em trong suốt quy trình học tập để em hoàn thành tốt việc học tập của bản thân.

Trong quá trình nghiên cứu và làm đề tài, do năng lực, kiến thức, trình độ bản thân em còn hạn hẹp nên không tránh khỏi những thiếu sót và em mong mọi nhận được sự thông cảm và những góp ý từ quý thầy cô cũng như các bạn trong lớp.

Em xin trân trọng cảm ơn!

Sinh viên thực hiện

Đào Trọng Hoàn

LỜI NÓI ĐẦU

Trong bối cảnh phát triển nhanh chóng của khoa học công nghệ, trí tuệ nhân tạo (AI) và học sâu (Deep Learning) đã và đang có những ứng dụng mạnh mẽ trong nhiều lĩnh vực khác nhau. Một trong những ứng dụng tiêu biểu và có nhiều tiềm năng là việc nhận diện và phân loại hình ảnh, đặc biệt là trong ngành y tế. Nhu cầu tự động hóa các quy trình phân loại ảnh y tế không chỉ giúp tăng khả năng dự đoán chính xác cho bác sĩ mà còn giúp bệnh nhân có thể chữa trị sớm căn bệnh góp phần vào sự phát triển bền vững của ngành.

Xuất phát từ những yêu cầu thực tiễn đó, đề án này tập trung nghiên cứu và ứng dụng mô hình học sâu DenseNet để giải quyết bài toán phân loại ảnh khối u não dựa trên hình ảnh. Qua quá trình nghiên cứu, triển khai và đánh giá, tôi hy vọng rằng kết quả đạt được sẽ mang lại những kiến thức hữu ích và có tính ứng dụng cao.

Đề án bao gồm các nội dung chính như khảo sát bài toán, lựa chọn phương pháp và các kỹ thuật giải quyết, cùng với việc huấn luyện và đánh giá mô hình. Kết quả của nghiên cứu không chỉ phục vụ cho bài toán phân loại ảnh y tế mà còn mở ra cơ hội cho các ứng dụng tương tự trong nhiều lĩnh vực khác.

Nội dung quyển báo cáo đề án tốt nghiệp sẽ bao gồm các chương như sau:

Chương 1: Khảo sát và phát biểu bài toán

Chương này nhằm giới thiệu tổng quan về bài toán phân loại ảnh y tế, đặc biệt là trong lĩnh vực chẩn đoán u não từ ảnh MRI. Trước hết, tôi trình bày vai trò quan trọng của các phương pháp hỗ trợ chẩn đoán hình ảnh trong y học hiện đại, cũng như tiềm năng ứng dụng của trí tuệ nhân tạo trong việc nâng cao hiệu quả và độ chính xác của quá trình chẩn đoán. Bên cạnh đó, chương này cũng đề cập đến bài toán nhận dạng ảnh u não và phát biểu bài toán.

Chương 2: Một số kỹ thuật giải quyết bài toán

Sau khi đã phát biểu và xác định rõ yêu cầu của bài toán trong Chương 1, chương này tập trung trình bày các kỹ thuật phổ biến hiện nay trong lĩnh vực nhận dạng ảnh y tế, đặc biệt là chẩn đoán u não từ ảnh MRI. Tôi tiến hành phân tích chi tiết từng phương pháp – từ các thuật toán truyền thống như phân cụm K-Means đến các mô hình học sâu hiện đại như mạng nơ-ron tích chập (CNN), ResNet50 và DenseNet121.

Đối với mỗi phương pháp, tôi đánh giá các ưu điểm và nhược điểm dựa trên khả năng trích xuất đặc trưng, hiệu suất phân loại cũng như tính khả thi khi áp dụng vào bài toán cụ thể. Từ những phân tích trên, tôi đề xuất giải pháp sử dụng mạng DenseNet121 – một kiến trúc mạng nơ-ron sâu có khả năng tái sử dụng đặc trưng hiệu quả – nhằm cải thiện độ chính xác phân loại và giảm thiểu hiện tượng mất thông tin trong quá trình huấn luyện. Giải pháp này kỳ vọng sẽ khắc phục được những hạn chế của các phương pháp trước và mang lại kết quả thực nghiệm khả quan hơn.

Chương 3: Thực nghiệm

Trong chương này, tôi trình bày chi tiết quá trình thực nghiệm áp dụng các kỹ thuật đã được giới thiệu ở Chương 2 để giải quyết bài toán nhận dạng u não từ ảnh MRI. Cụ thể, tôi tiến hành huấn luyện và đánh giá mô hình DenseNet121 trên tập dữ liệu thực tế, đồng thời thực hiện so sánh kết quả với mô hình ResNet50 – một kiến trúc phổ biến và hiệu quả trong các bài toán phân loại ảnh. Các chỉ số đánh giá như độ chính xác, độ nhạy, độ đặc hiệu và F1-score được sử dụng nhằm đảm bảo tính khách quan trong việc so sánh hiệu suất giữa các mô hình. Thông qua quá trình phân tích kết quả, tôi đưa ra những nhận xét về ưu điểm và hạn chế của từng phương pháp, từ đó rút ra mô hình phù hợp nhất cho việc triển khai hệ thống chẩn đoán u não tự động.

Chương 4: Cài đặt chương trình và kiểm thử

Dựa trên kết quả thực nghiệm thu được từ quá trình huấn luyện mô hình DenseNet121, tôi tiến hành xây dựng một sản phẩm demo dưới dạng trang web nhằm hỗ trợ nhận dạng các loại u não từ ảnh MRI. Ứng dụng này sử dụng mô hình đã được huấn luyện để thực hiện dự đoán, phục vụ nhu cầu kiểm thử và đánh giá hiệu quả mô hình trong môi trường thực tế. Phần back-end của hệ thống được xây dựng bằng thư viện Flask, giúp triển khai mô hình học sâu và xử lý logic phía máy chủ. Phần giao diện người dùng (front-end) được thiết kế bằng Bootstrap, kết hợp với CSS và JavaScript nhằm tạo ra một trải nghiệm thân thiện, dễ sử dụng. Ngoài ra, hệ thống còn tích hợp chức năng lưu trữ kết quả chẩn đoán vào cơ sở dữ liệu, từ đó hỗ trợ cho việc thống kê, đánh giá và cải tiến mô hình trong các giai đoạn phát triển tiếp theo.

Phần kết

Trong phần kết luận, tôi đã tổng hợp lại toàn bộ quá trình nghiên cứu và triển khai đề tài "Nghiên cứu và phát triển ứng dụng nhận dạng u não từ ảnh MRI bằng mạng DenseNet". Thông qua việc khảo sát thực trạng, lựa chọn hướng tiếp cận phù hợp và triển khai mô hình DenseNet121 trên tập dữ liệu ảnh MRI, tôi đã xây dựng được một hệ thống có khả năng hỗ trợ chẩn đoán u não tự động với độ chính xác đáng khích lệ.

Kết quả thực nghiệm cho thấy mô hình DenseNet121 mang lại hiệu suất cao hơn so với một số mô hình phổ biến khác như ResNet50 trong cùng điều kiện huấn luyện. Đồng thời, tôi cũng đã phát triển một ứng dụng web demo giúp minh họa khả năng ứng dụng mô hình vào thực tiễn.

Tuy nhiên, hệ thống hiện tại vẫn còn một số hạn chế như chưa thực hiện phân đoạn khối u, độ chính xác ở một số loại u còn chưa cao, và chưa được thử nghiệm trong môi trường lâm sàng thực tế.

Trong tương lai, tôi dự định tiếp tục cải tiến mô hình bằng cách tích hợp thêm các kỹ thuật nâng cao như attention mechanism, thử nghiệm các mô hình kiến trúc transformer hiện đại, cũng như mở rộng bộ dữ liệu huấn luyện để tăng độ tổng quát. Đồng thời, tôi sẽ hoàn thiện hệ thống web, bổ sung tính năng và hướng tới việc triển khai thực tế trong các cơ sở y tế.

CHƯƠNG 1: KHẢO SÁT VÀ PHÁT BIỂU BÀI TOÁN

1.1 Khảo sát hiện trạng y tế Việt Nam

1.1.1 Hiện trạng y tế Việt Nam

Hiện trạng y tế ở Việt Nam hiện nay có nhiều điểm sáng đáng ghi nhận. Hệ thống y tế bốn cấp (trung ương – tỉnh – huyện – xã) được duy trì rộng khắp cả nước, giúp người dân ở cả thành thị và nông thôn tiếp cận dịch vụ y tế cơ bản. Tỷ lệ người dân tham gia bảo hiểm y tế đã đạt khoảng 92% (tính đến năm 2024)[1], góp phần giảm gánh nặng chi phí khám chữa bệnh. Năng lực chuyên môn của ngành y tế cũng ngày càng được nâng cao, thể hiện qua việc nhiều bệnh viện lớn như Bạch Mai, Chợ Rẫy, 108... thực hiện thành công các kỹ thuật y khoa cao như ghép tạng, can thiệp tim mạch, xạ trị ung thư. Ngoài ra, công tác kiểm soát dịch bệnh, đặc biệt trong đại dịch COVID-19, đã cho thấy khả năng ứng phó nhanh nhạy và hiệu quả của hệ thống y tế Việt Nam.

Tuy nhiên, bên cạnh những thành tựu, ngành y tế Việt Nam vẫn đối mặt với không ít thách thức. Tình trạng quá tải bệnh viện, đặc biệt tại các bệnh viện lớn ở Hà Nội và TP.HCM, vẫn diễn ra thường xuyên, khiến bệnh nhân phải chờ đợi lâu và đôi khi phải nằm ghép giường. Sự chênh lệch về chất lượng dịch vụ y tế giữa thành thị và vùng sâu, vùng xa còn lớn, với việc thiếu thốn bác sĩ, trang thiết bị, và thuốc men ở nhiều địa phương miền núi, Tây Nguyên. Tuyến y tế cơ sở vẫn còn hạn chế, dẫn đến việc người dân thường xuyên vượt tuyến lên các bệnh viện trung ương. Bên cạnh đó, mặc dù có bảo hiểm y tế, nhiều kỹ thuật cao và thuốc đặc trị vẫn phải chi trả ngoài, gây khó khăn tài chính cho người bệnh. Ngành y tế cũng đối mặt với tình trạng thiếu nhân lực, khi nhiều bác sĩ giỏi chuyển sang khu vực tư nhân hoặc ra nước ngoài làm việc.

Trước những thách thức đó, ngành y tế Việt Nam đang định hình nhiều xu hướng phát triển mới. Chuyển đổi số trong y tế được thúc đẩy mạnh mẽ với các ứng dụng như bệnh án điện tử, tư vấn khám bệnh từ xa (telemedicine), và quản lý hồ sơ sức khỏe cá nhân. Y tế dự phòng cũng được chú trọng hơn nhằm phòng

chống các bệnh không lây nhiễm như ung thư, tiểu đường, tim mạch. Bên cạnh hệ thống y tế công lập, khu vực y tế tư nhân đang phát triển nhanh chóng, cung cấp dịch vụ chăm sóc sức khỏe cao cấp cho người dân. Ngoài ra, Việt Nam cũng đang tăng cường hợp tác quốc tế trong nghiên cứu khoa học và đào tạo nhân lực y tế, nhằm nâng cao chất lượng chuyên môn và hội nhập sâu hơn với thế giới.

1.1.2 Ứng dụng công nghệ cao chuẩn đoán bệnh ở việt nam

Trong những năm gần đây, ứng dụng công nghệ cao trong chẩn đoán bệnh tại Việt Nam đã có bước tiến mạnh mẽ. Nhiều bệnh viện lớn như Bệnh viện Bạch Mai, Bệnh viện Chợ Rẫy, Bệnh viện Trung ương Quân đội 108... đã triển khai các thiết bị y tế hiện đại như máy chụp cộng hưởng từ (MRI) 3.0 Tesla, máy chụp cắt lớp vi tính (CT-Scan) đa lát cắt, hệ thống PET/CT trong chẩn đoán ung thư, và máy siêu âm đàn hồi mô. Các kỹ thuật tiên tiến này giúp phát hiện sớm các tổn thương nhỏ, hỗ trợ bác sĩ xác định bệnh lý với độ chính xác cao hơn, từ đó nâng cao hiệu quả điều trị cho người bệnh.

Bên cạnh việc trang bị máy móc hiện đại, Việt Nam cũng đang đẩy mạnh ứng dụng trí tuệ nhân tạo (AI) trong hỗ trợ chẩn đoán. Một số bệnh viện đã thử nghiệm hệ thống AI đọc phim X-quang phổi để phát hiện lao và ung thư phổi sớm. Ngoài ra, AI còn được sử dụng trong phân tích hình ảnh MRI não để hỗ trợ chẩn đoán đột quỵ, u não, và các bệnh lý thần kinh. Những ứng dụng này giúp rút ngắn thời gian đọc kết quả, giảm tải cho bác sĩ chẩn đoán hình ảnh và tăng khả năng phát hiện bệnh ở giai đoạn sớm.

Ứng dụng công nghệ thông tin trong quản lý dữ liệu bệnh nhân cũng góp phần cải thiện quy trình chẩn đoán. Nhiều bệnh viện đã triển khai hệ thống bệnh án điện tử, lưu trữ hình ảnh PACS (Picture Archiving and Communication System), cho phép bác sĩ truy cập hồ sơ bệnh nhân một cách nhanh chóng, hỗ trợ chẩn đoán và theo dõi điều trị liên tục. Khám bệnh từ xa (telemedicine) cũng bắt đầu được triển khai mạnh mẽ, đặc biệt ở các tỉnh vùng sâu vùng xa, cho phép bệnh

nhân được tư vấn và chẩn đoán bởi các chuyên gia đầu ngành mà không cần di chuyển.

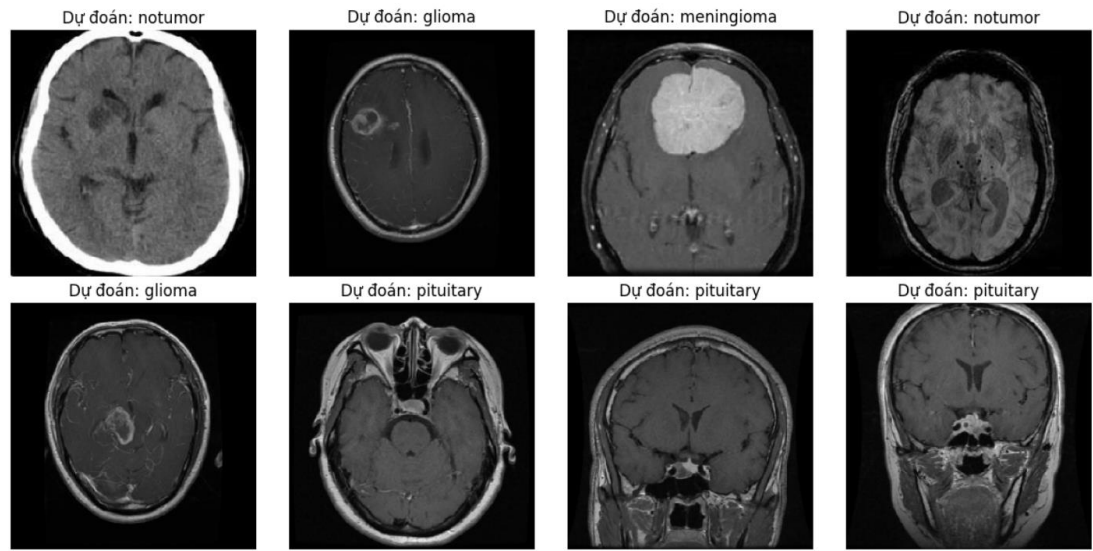
Mặc dù đạt được nhiều thành tựu, việc ứng dụng công nghệ cao trong chẩn đoán bệnh tại Việt Nam vẫn còn một số hạn chế. Chi phí đầu tư ban đầu cho thiết bị công nghệ cao còn lớn, khiến việc phổ cập ra toàn bộ hệ thống y tế cơ sở gặp khó khăn. Ngoài ra, việc thiếu hụt nhân lực y tế có khả năng vận hành và khai thác tối ưu các công nghệ mới cũng là một thách thức cần được khắc phục trong thời gian tới.

Nhìn chung, ứng dụng công nghệ cao trong chẩn đoán bệnh tại Việt Nam đang từng bước phát triển, mang lại hy vọng cải thiện chất lượng chăm sóc sức khỏe cho người dân. Với đà đầu tư và đổi mới hiện nay, trong tương lai, việc chuẩn đoán bệnh sớm, chính xác và ít xâm lấn sẽ trở thành tiêu chuẩn phổ biến tại nhiều cơ sở y tế trên toàn quốc.

1.2 Bài toán nhận dạng ảnh u não

1.2.1 Khái quát về u não

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 phổi, thận...) rồi theo máu đến não, được gọi là u di căn não. Cơ chế hình thành nên u não thông thường là từ lúc sinh ra đến lúc mất đi, không có thêm tế bào thần kinh nào được sinh thêm ra nữa. Khi có đột biến không rõ nguyên nhân trong DNA khiến các tế bào phân chia mất kiểm soát thì sẽ hình thành nên u não. Tốc độ phát triển cũng như vị trí của u não quyết định mức độ nghiêm trọng và tầm ảnh hưởng của nó đến chức năng hệ thần kinh, thậm chí là đe dọa đến tính mạng nếu không được chẩn đoán, theo dõi và chữa trị kịp thời.

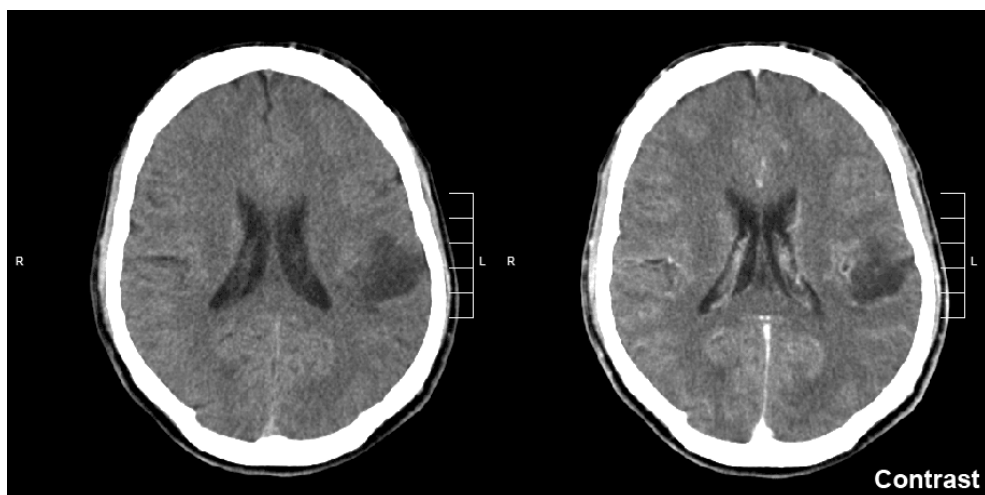


Hình 1.1 Ảnh u não

U não chiếm 2% trong tổng số các ca ung thư từ mọi nhóm tuổi. Trong số các trường hợp tử vong do ung thư ở nhóm trẻ em dưới 15 tuổi và nhóm từ 20-39 tuổi, bệnh u não là nguyên nhân gây tử vong cao thứ 2. Người ngoài 85 tuổi có tỉ lệ bị u não cao nhất.

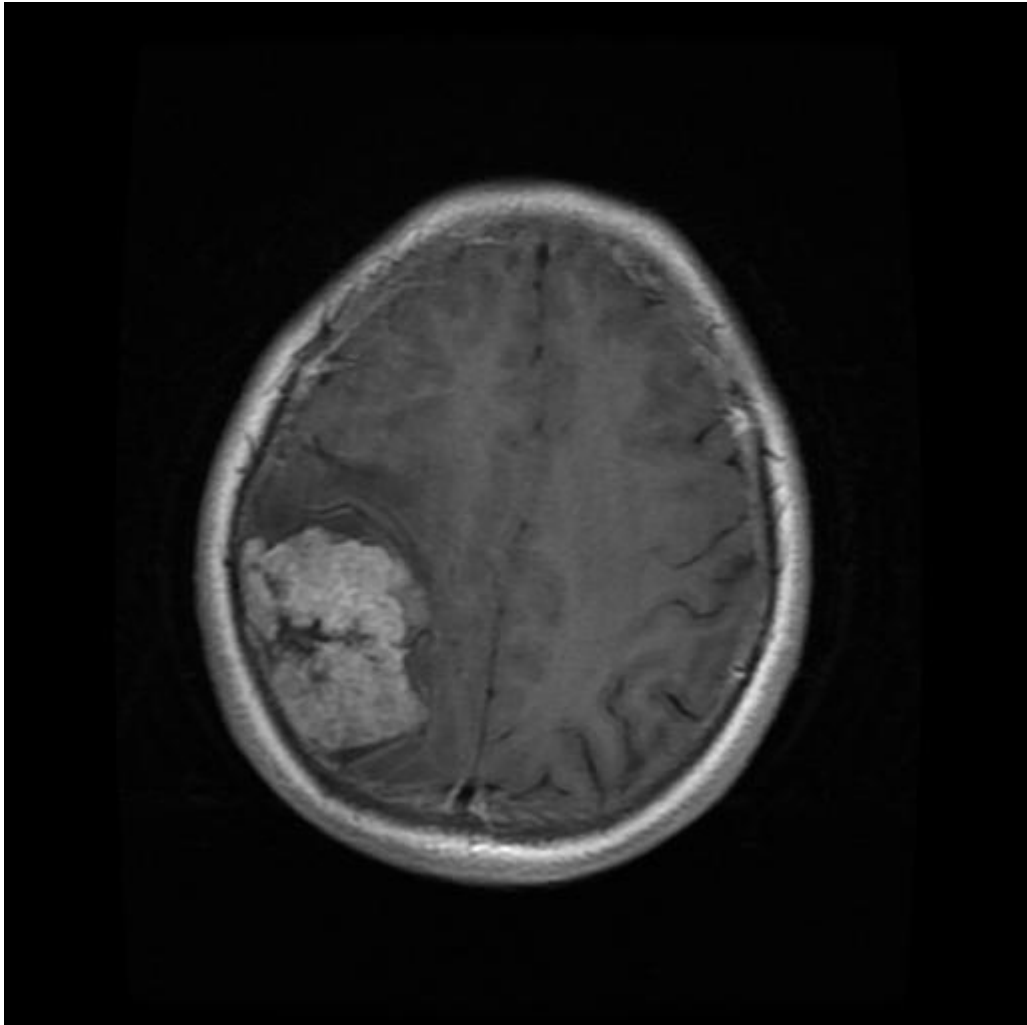
Hiện nay, có 04 loại khối u não thường gặp là:

- U thần kinh đệm (Gliomas): Đây là khối u não bắt đầu từ trong các tế bào thần kinh đệm ở não hoặc tủy sống. U não Gliomas là loại u não nguyên phát ác tính, chiếm khoảng 50,4% tổng số các ca u não và 78% các ca có khối u não nguyên phát ác tính.



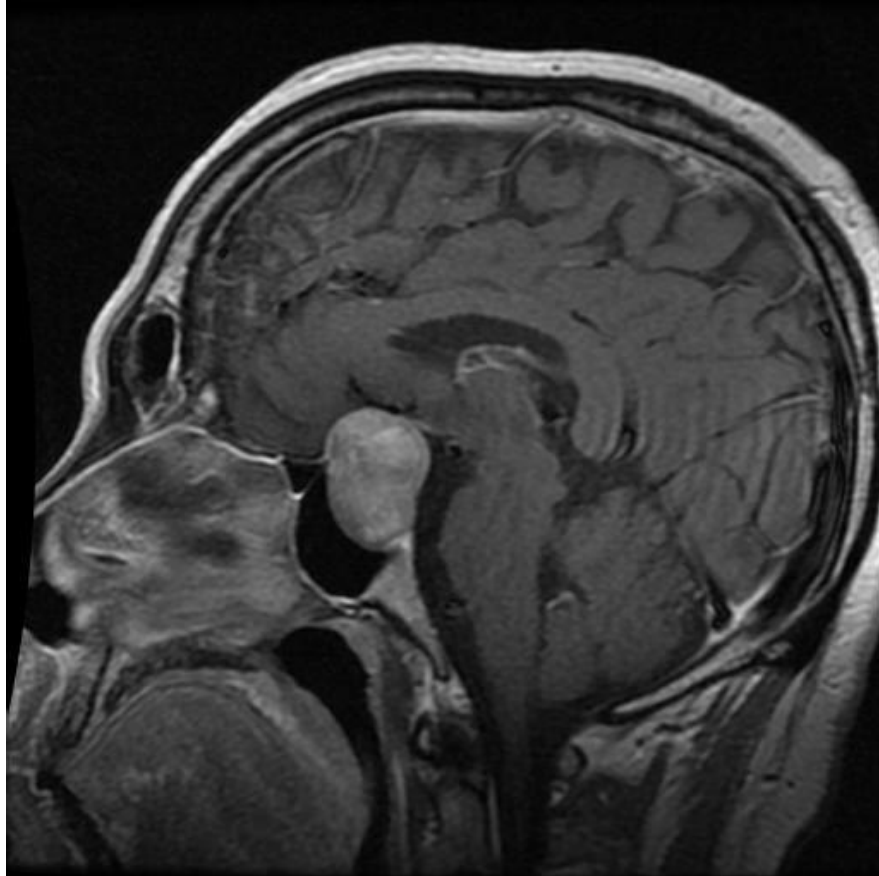
Hình 1.2 U thần kinh đệm

- U màng não(Meningioma): Là một khối u phát triển chậm, hình thành từ màng não hay lớp màng bao quanh tủy sống. U màng não là loại u lành tính, thường xuất hiện ở nữ giới, chiếm khoảng 20,8% tổng số các ca bị u não với tỉ lệ tái phát sau phẫu thuật thấp (ít hơn 20%).



Hình 1.3 U màng não

- U tuyến yên(Pituitary): Là khối u xảy ra trong tuyến yên (nằm ở bề mặt dưới não) với hơn 60% các ca được chẩn đoán là lành tính, 35% là loại khối u có xâm lấn. Theo thống kê, u tuyến yên chiếm từ 10% – 25% trên tổng số các ca u não và tỷ lệ mắc bệnh có thể lên đến 17% dân số.



Hình 1.4 U tuyến yên

- Khối u thần kinh ngoại biên: Do các nguyên bào sợi tăng trưởng bao quanh bó thần kinh gây ra và chiếm khoảng 10% tổng số ca bị u não. Hầu hết các khối u thần kinh ngoại biên là lành tính (không phải ung thư). Tuy nhiên, khối u chèn ép thần kinh gây đau và có thể làm mất khả năng kiểm soát cơ bắp.

Nguyên nhân gây u não chính xác trong hầu hết mọi trường hợp là không thể xác định. Tuy nhiên, nguyên nhân tiềm ẩn thì có rất nhiều. Bất cứ điều gì làm tăng khả năng mắc bệnh u não đều là có thể được xem là một yếu tố nguy cơ của bệnh (tức nguyên nhân tiềm ẩn). Việc bạn có một trong số các nguy cơ gây u não

dưới đây không chắc chắn là bạn sẽ bị u não trong tương lai. Các yếu tố nguy cơ làm tăng rủi ro u não có thể là:

- Tuổi tác: Người càng lớn tuổi càng có nguy cơ bị u não. Hầu hết các khối u não xảy ra ở người lớn tuổi từ 85 đến 89, mặc dù vẫn có một số loại u não phổ biến hơn ở trẻ em dưới 15 tuổi.
- Tiền sử gia đình (di truyền): Theo báo cáo, chỉ có từ 5-10% tổng số ca ung thư là do di truyền. U não chỉ chiếm 2% tổng số ca ung thư trên toàn thế giới, do đó tỉ lệ khối u não được di truyền là rất thấp. Một số tình trạng di truyền được biết là làm tăng nguy cơ mắc khối u não, bao gồm: bệnh xơ cứng củ, bệnh u sợi thần kinh loại 1, loại 2, hội chứng Turner, hội chứng Li-Fraumeni, hội chứng Turcot, hội chứng Gorlin,...
- Chế độ ăn thiếu khoa học: Một số nghiên cứu đã chỉ ra rằng các hợp chất N-nitroso trong chế độ ăn uống có thể ảnh hưởng đến nguy cơ mắc các khối u não ở trẻ em và người lớn. Gần đây, Tiến sĩ. Lee Wrensch phát hiện ra rằng người mắc bệnh u thần kinh đệm có tỉ lệ lớn tiêu thụ chế độ ăn ít trái cây, ít rau quả, ít vitamin C mà chứa nhiều nitrit như phô mai, cá, thịt xông khói, thức ăn đã qua chế biến, lên men, ử muối qua đêm (cá khô), đồ đóng hộp.
- Thừa cân và béo phì: Thừa cân hoặc béo phì làm tăng nguy cơ mắc bệnh u màng não. Khoảng 2% tổng số ca được chẩn đoán u não ở Anh mỗi năm là do thừa cân hoặc béo phì. Thừa cân hoặc béo phì làm tăng nguy cơ mắc bệnh u màng não. Khoảng 2% tổng số ca được chẩn đoán u não ở Anh mỗi năm là do thừa cân hoặc béo phì.
- Phơi nhiễm hóa chất: Một số ngành nghề do môi trường làm việc đặc thù cần tiếp xúc với nhiều hóa chất có thể làm tăng nguy cơ ung thư não, chẳng hạn như: Người làm nông nghiệp phải tiếp xúc nhiều với thuốc trừ sâu, Công nhân làm việc trong môi trường nhiều kim loại nặng (niken, thủy ngân),...

- Tiếp xúc với bức xạ: Bức xạ ion hóa là một loại bức xạ được sử dụng bởi một số quy trình quét y tế, chẳng hạn như chụp X-quang và chụp CT. Những người đã tiếp xúc với bức xạ ion hóa có nguy cơ mắc các khối u não cao hơn người bình thường. Do đó, nếu bạn đã có tiền sử xạ trị trước đây với các bệnh ung thư khác thì cũng có thể làm tăng nguy cơ bị u não của bạn lên một chút. Tuy nhiên, u não do tiếp xúc với bức xạ xảy ra với tỉ lệ rất hiếm (dưới 1%).

Khối u não rất nguy hiểm dù là u não lành tính hay u não ác tính. Bệnh dù được điều trị kịp thời hay không đều có thể dễ dàng để lại những biến chứng nhất định, ảnh hưởng nghiêm trọng đến sinh hoạt, có thể rút ngắn tuổi thọ bệnh nhân hay thậm chí đe dọa tính mạng.

1.2.2 Hiện trạng u não ở Việt Nam

Bệnh viện Hữu nghị Việt Đức: Mỗi năm tiếp nhận và điều trị hơn 2.500 ca u não, bao gồm cả u lành tính và ác tính [2].

Globocan 2022: Ghi nhận 2.829 ca mắc mới và 2.431 ca tử vong do ung thư não và hệ thần kinh tại Việt Nam, xếp thứ 14 về tỷ lệ mắc và thứ 11 về tỷ lệ tử vong trong các loại ung thư [3].

Bệnh viện Ung bướu TP.HCM: Thống kê cho thấy u não chiếm khoảng 2% trong tổng số các loại ung thư, tuy nhiên nhiều trường hợp được phát hiện muộn, dẫn đến khó khăn trong điều trị [4].

1.3 Phát Biểu Bài Toán

Bài toán đặt ra là xây dựng một ứng dụng nhận dạng u não từ ảnh MRI bằng cách sử dụng mạng DenseNet. Ứng dụng này có thể hỗ trợ bác sĩ trong việc chẩn đoán nhanh và chính xác sự hiện diện của khối u trong não, cũng như phân loại loại u dựa trên hình ảnh MRI.

1.3.1 Đầu vào

Ảnh đầu vào:

- **Dạng ảnh:** Ảnh số chứa hình ảnh MRI của một bộ não duy nhất. Các ảnh có thể là ảnh chụp hoặc ảnh đã được xử lý.
- **Độ phân giải:** Ảnh phải có kích thước cố định là 224 x 224 pixels
- **Màu sắc:** Ảnh phải là ảnh RGB với 3 kênh màu (Đỏ, Xanh lá, Xanh dương).
Chuẩn hóa: Ảnh đầu vào phải được chuẩn hóa về giá trị pixel. Thông thường, các giá trị pixel nằm trong khoảng $[0, 1]$ sau khi chia giá trị pixel gốc (0-255) cho 255.
- **Định dạng ảnh:** Có thể là các định dạng phổ biến như JPEG hoặc PNG.

Đặc điểm của ảnh:

- **Một đối tượng duy nhất:** Mỗi ảnh chỉ chứa một bộ não duy nhất, không có vật cản lớn che khuất đối tượng hoặc các đối tượng khác gây nhiễu.
- **Đủ dữ liệu mẫu:** Đảm bảo có đủ ảnh đại diện cho từng loại u não để tránh mất cân bằng dữ liệu.
- **Góc nhìn đa dạng:** Ảnh của các loại u não cần bao gồm nhiều góc chụp khác nhau (trực diện, nghiêng) để mô hình có thể học các đặc trưng đặc thù của từng loại u não từ nhiều góc độ khác nhau.

1.3.2 Đầu ra

Dạng đầu ra:

- **Vector xác suất:** Đầu ra từ mô hình là một vector xác suất, mỗi phần tử của vector tương ứng với một loại u não. Số lượng phần tử của vector bằng với số loại u não cần phân loại là 4(bao gồm cả không có khối u).
- **Xác suất chuẩn hóa:** Tổng các xác suất trong vector luôn bằng 1, sử dụng hàm softmax cho lớp đầu ra để chuẩn hóa xác suất này.

Nhãn dự đoán:

- Lựa chọn nhãn có xác suất cao nhất: Loại u não tương ứng với xác suất cao nhất trong vector sẽ được chọn làm nhãn dự đoán cho ảnh.
- Độ chính xác của nhãn: Để đánh giá hiệu quả mô hình, có thể yêu cầu độ chính xác trên tập kiểm tra phải đạt ngưỡng xác định, khoảng trên 90%.
- Thông tin bổ sung: Ngoài nhãn dự đoán, mô hình có thể xuất ra các thông tin bổ sung như mức độ tự tin (confidence score) để người dùng hiểu mức độ chắc chắn của mô hình với dự đoán đó.

1.3.3 Các ràng buộc của bài toán

Ràng buộc về đầu vào:

- Độ rõ của ảnh: Ảnh đầu vào phải rõ ràng, không bị nhòe hoặc nhiễu mạnh vì điều này có thể làm giảm độ chính xác của mô hình. Các yếu tố ánh sáng phải hợp lý để không che khuất đặc điểm nhận dạng của khối u.
- Định dạng ảnh cố định: Đảm bảo rằng tất cả ảnh đầu vào được chuẩn hóa về kích thước và định dạng trước khi đưa vào mô hình. Sử dụng cùng kích thước ảnh giúp duy trì tính nhất quán và tối ưu hóa hiệu suất mô hình.
- Chất lượng dữ liệu: Cần có một lượng lớn ảnh chất lượng cao cho từng loại u não. Điều này đặc biệt quan trọng vì kết quả để phục vụ cho y học, lĩnh vực không nên có sai số.
- Tiền xử lý dữ liệu: Ngoài việc chuẩn hóa pixel, các bước tiền xử lý khác như làm mờ ảnh, xoay ảnh, cắt ảnh để tăng tính đa dạng trong dữ liệu huấn luyện (data augmentation) nhằm cải thiện khả năng tổng quát hóa của mô hình.

Ràng buộc về đầu ra:

- Yêu cầu độ chính xác cao: Mô hình phải đạt độ chính xác cao trên tập kiểm tra (>90%) để đáp ứng yêu cầu của bài toán phân loại.

- Khả năng phân biệt các khối u não chính xác: Mô hình cần có khả năng phân biệt giữa các khối u não hoặc là không có khối u một cách chính xác để có tính ứng dụng.

Ràng buộc về môi trường

- Phần cứng: Để huấn luyện mô hình CNN hiệu quả, cần có GPU hoặc TPU để tăng tốc độ tính toán, đặc biệt với mô hình lớn và dữ liệu nhiều ảnh.
- Phần mềm: Môi trường lập trình Python cùng với các thư viện học sâu như TensorFlow hoặc PyTorch là cần thiết để xây dựng và huấn luyện mô hình.

Ràng buộc về hiệu năng

- Tốc độ dự đoán: Mô hình cần dự đoán nhanh (thời gian suy luận dưới 300ms mỗi ảnh đầu vào) để có thể tích hợp vào các ứng dụng thời gian thực hoặc các hệ thống phân loại tự động.
- Tối ưu hóa mô hình: Sử dụng các kỹ thuật giảm kích thước mô hình như chuyển đổi sang TensorRT hoặc sử dụng phương pháp pruning và quantization để giảm kích thước và tăng tốc độ mô hình mà không làm mất nhiều độ chính xác.
- Khả năng mở rộng: Đảm bảo mô hình có thể dễ dàng mở rộng để phân loại thêm loại u não mới nếu cần, mà không cần huấn luyện lại từ đầu.
- Khả năng tổng quát hóa: Mô hình cần được huấn luyện đủ lâu để tránh quá khớp (overfitting) và đảm bảo tính tổng quát khi phân loại ảnh chưa từng gặp trong tập huấn luyện.

1.3.4 Mục tiêu của bài toán

Mục tiêu của bài toán này là xây dựng một mô hình trí tuệ nhân tạo sử dụng mạng nơ-ron tích chập (CNN), cụ thể là DenseNet, để tự động phân loại khối u não dựa trên ảnh chụp MRI. Hình ảnh MRI là một hình ảnh phổ biến trong y học, giúp cung cấp thông tin chi tiết về cấu trúc não bộ. Tuy nhiên, việc phân tích các ảnh này thường đòi hỏi sự can thiệp của các chuyên gia y tế có kinh nghiệm, dẫn

đến sự phụ thuộc vào yếu tố con người và có thể xảy ra sai sót. Do đó, việc ứng dụng trí tuệ nhân tạo nhằm tự động hóa và nâng cao độ chính xác trong chẩn đoán là một hướng nghiên cứu quan trọng.

Trong nghiên cứu này, mô hình sẽ được huấn luyện để nhận diện và phân loại các khối u như Glioma, Meningioma, Pituitary Tumor hoặc xác định trường hợp không có khối u (No tumor). DenseNet là một kiến trúc CNN tiên tiến, có khả năng trích xuất đặc trưng hiệu quả từ hình ảnh y tế nhờ vào thiết kế sâu và tối ưu hóa việc xử lý dữ liệu. Việc sử dụng mô hình này giúp cải thiện độ chính xác trong chẩn đoán, đồng thời giảm thời gian phân tích hình ảnh so với các phương pháp truyền thống.

Bên cạnh đó, nghiên cứu cũng tập trung vào việc tối ưu hóa mô hình để đạt hiệu suất cao, tránh overfitting và đảm bảo khả năng tổng quát hóa trên tập dữ liệu mới. Điều này bao gồm các kỹ thuật như tiền xử lý dữ liệu, tinh chỉnh mô hình (fine-tuning) và đánh giá các chỉ số trên bộ dữ liệu. Việc đảm bảo mô hình hoạt động ổn định trên dữ liệu thực tế sẽ giúp nâng cao tính ứng dụng trong môi trường lâm sàng.

Kết quả của bài toán có thể hỗ trợ bác sĩ trong việc phát hiện sớm và phân loại khối u não, góp phần nâng cao hiệu quả chẩn đoán và điều trị cho bệnh nhân. Một hệ thống phân loại chính xác sẽ giúp giảm gánh nặng công việc cho bác sĩ, hỗ trợ đưa ra quyết định nhanh chóng và chính xác hơn, từ đó cải thiện chất lượng chăm sóc y tế.

1.3.5 Cơ hội và thách thức

a) Cơ hội

Hỗ trợ chẩn đoán y tế: Ứng dụng giúp bác sĩ phát hiện sớm và phân loại khối u não chính xác hơn, từ đó hỗ trợ đưa ra quyết định điều trị kịp thời.

Tăng độ chính xác và tốc độ: Mô hình DenseNet có khả năng học các đặc trưng phức tạp trong ảnh MRI, giúp cải thiện độ chính xác so với phương pháp truyền thống.

Ứng dụng trong hệ thống y tế thông minh: Có thể tích hợp vào phần mềm hỗ trợ chẩn đoán bệnh, bệnh viện số hoặc hệ thống AI y tế.

Tự động hóa và giảm tải cho bác sĩ: Giúp bác sĩ tiết kiệm thời gian trong việc phân tích hình ảnh MRI, tập trung vào các trường hợp quan trọng.

Tiềm năng mở rộng: Hệ thống có thể được cải thiện để nhận dạng nhiều loại bệnh lý não khác, không chỉ u não.

b) Thách thức

Chất lượng dữ liệu ảnh MRI: Ảnh có thể bị nhiễu, độ phân giải khác nhau, hoặc bị ảnh hưởng bởi thiết bị chụp, gây khó khăn trong việc huấn luyện mô hình.

Sự tương đồng giữa các loại khối u: Một số loại u có hình dạng khá giống nhau, khiến mô hình dễ bị nhầm lẫn.

Thiếu tập dữ liệu đủ lớn và đa dạng: Để huấn luyện một mô hình AI hiệu quả, cần một tập dữ liệu phong phú từ nhiều nguồn bệnh viện khác nhau.

Khả năng tổng quát hóa: Mô hình có thể hoạt động tốt trên tập huấn luyện nhưng gặp khó khăn khi áp dụng lên ảnh MRI từ các bệnh viện khác do sự khác biệt về dữ liệu.

Tính pháp lý và đạo đức: Việc ứng dụng AI trong y tế đòi hỏi tuân thủ các quy định về bảo mật dữ liệu và đạo đức y khoa, đảm bảo rằng AI chỉ đóng vai trò hỗ trợ, không thay thế bác sĩ.

1.4 Tóm tắt chương

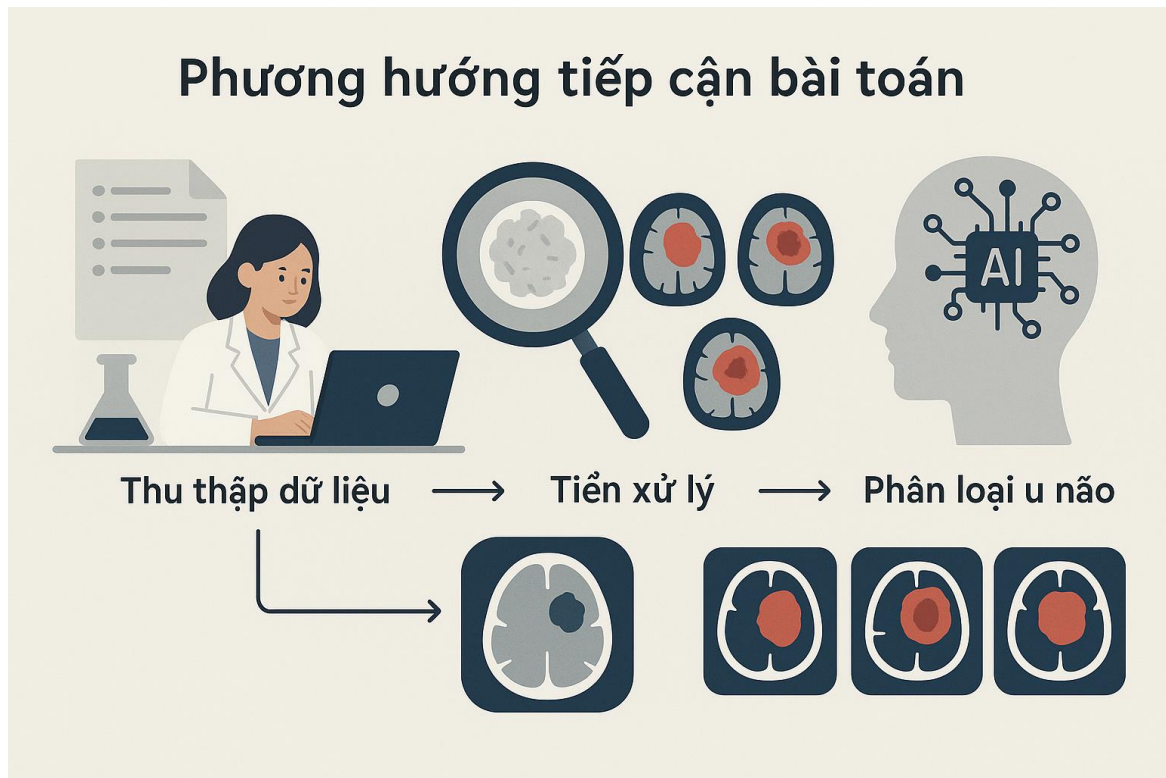
Trong chương 1, tôi đã tiến hành khảo sát hiện trạng ngành y tế tại Việt Nam, đặc biệt là những khó khăn, thách thức và xu hướng ứng dụng công nghệ cao trong công tác chẩn đoán bệnh. Tiếp theo, chương trình bày tổng quan về bệnh

u não – một trong những căn bệnh nguy hiểm nhưng khó phát hiện sớm – cũng như thực trạng u não tại Việt Nam. Dựa trên cơ sở đó, bài toán nhận dạng ảnh u não được phát biểu rõ ràng, bao gồm đầu vào, đầu ra, các ràng buộc và mục tiêu cụ thể. Cuối cùng, chương cũng nêu bật những cơ hội và thách thức trong việc ứng dụng trí tuệ nhân tạo vào chẩn đoán hình ảnh y học, đặc biệt trong bối cảnh thực tiễn tại Việt Nam. Phần nội dung này đặt nền móng quan trọng cho các chương tiếp theo, nơi sẽ trình bày các phương pháp tiếp cận, mô hình đề xuất và kết quả thực nghiệm.

CHƯƠNG 2: CÁC KỸ THUẬT GIẢI QUYẾT BÀI TOÁN

2.1 Phương hướng tiếp cận bài toán

Quá trình giải quyết bài toán bắt đầu bằng việc thu thập và tiền xử lý các bộ dữ liệu thực nghiệm liên quan tới bệnh u não, đặc biệt tập trung vào 3 loại u não phổ biến gây ra nhiều khó khăn trong việc nhận biết và điều trị. Tiếp đến là nghiên cứu và áp dụng kỹ thuật trí tuệ nhân tạo để nhận dạng và phân loại u não.



Hình 2.1 Hình ảnh các khối u với nhãn

2.2 Một số kỹ thuật giải quyết bài toán

Phần lớn các kỹ thuật giải quyết bài toán phân loại ảnh u não thường sử dụng nhãn cho từng loại u não. Các kỹ thuật này có thể chia thành 3 phương pháp dựa trên tính khả dụng cho các nhãn là:

- Nhận dạng u não bằng kỹ thuật học có giám sát (Supervised Learning): là thuật toán dự đoán đầu ra (outcome) của một dữ liệu mới (new input) dựa trên các cặp (input, outcome) đã biết từ trước. Cặp dữ liệu này còn

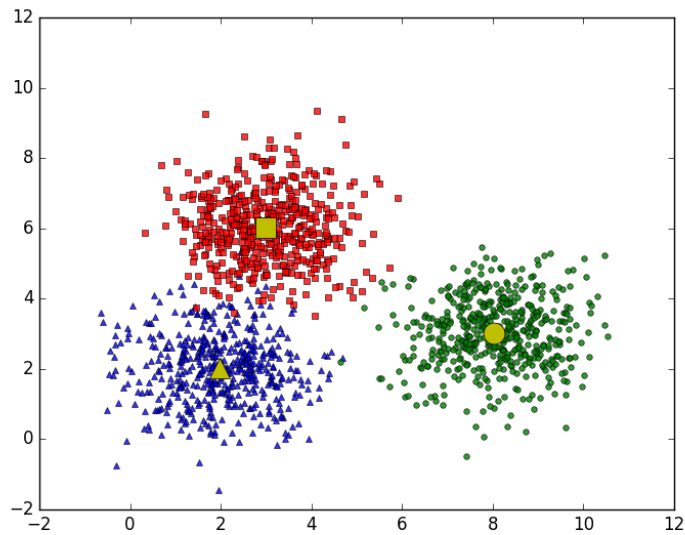
được gọi là (data, label), tức (dữ liệu, nhãn). Supervised learning là nhóm phổ biến nhất trong các thuật toán Machine Learning.

- Nhận dạng u não bằng kỹ thuật học không giám sát (Unsupervised Learning): Trong thuật toán này, chúng ta không biết được outcome hay nhãn mà chỉ có dữ liệu đầu vào. Thuật toán unsupervised learning sẽ dựa vào cấu trúc của dữ liệu để thực hiện một công việc nào đó, ví dụ như phân nhóm (clustering) hoặc giảm số chiều của dữ liệu (dimension reduction) để thuận tiện trong việc lưu trữ và tính toán. Một cách toán học, Unsupervised learning là khi chúng ta chỉ có dữ liệu vào X mà không biết nhãn Y tương ứng.
- Nhận dạng u não bằng kỹ thuật học bán giám sát (Semi- Supervised Learning): Các bài toán khi chúng ta có một lượng lớn dữ liệu X nhưng chỉ một phần trong chúng được gán nhãn được gọi là Semi- Supervised Learning. Những bài toán thuộc nhóm này nằm giữa hai nhóm Supervised Learning và Unsupervised Learning.

2.3 K-Means Clustering

2.3.1 Định nghĩa

- K-Means là một thuật toán phân cụm (clustering) không giám sát, giúp chia dữ liệu thành K cụm (clusters) khác nhau dựa trên độ tương đồng.
- Không yêu cầu nhãn (label) sẵn có.



Hình 2.2 Mô hình K-Means

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[5])

2.3.2 Cách hoạt động

Quy trình gồm 4 bước lặp:

- Khởi tạo: Chọn ngẫu nhiên K điểm làm tâm cụm (centroids).
- Gán cụm: Gán mỗi điểm dữ liệu vào cụm có tâm gần nhất (theo khoảng cách, thường là khoảng cách Euclidean).
- Cập nhật tâm cụm: Tính lại trung bình các điểm trong mỗi cụm để làm tâm cụm mới.
- Lặp lại bước 2-3 cho đến khi:
 - Các tâm cụm không thay đổi nhiều (hội tụ).
 - Hoặc đạt số vòng lặp tối đa.

2.3.3 Công thức

Khoảng cách Euclid thường dùng:

$$d(x,y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \dots + (x_n - y_n)^2}$$

Hàm mục tiêu muốn tối thiểu tổng bình phương khoảng cách trong cụm:

$$SSE = \sum_{i=1}^K \sum_{x \in C_i} \|x - \mu_i\|^2$$

với:

- K là số cụm
- C_i là cụm thứ i
- μ_i là tâm cụm i

2.3.4 Ưu điểm

- Dễ hiểu, dễ cài đặt.
- Tính toán nhanh với dữ liệu lớn.
- Tốt khi các cụm có hình cầu và kích thước tương đương.

2.3.5 Nhược điểm

- Phải chọn trước K (số cụm).
- Nhạy cảm với điểm khởi tạo (dễ kẹt ở nghiệm cục bộ).
- Không phù hợp cho các cụm có hình dạng phức tạp (không tròn).
- Nhạy cảm với outlier (dữ liệu ngoại lai).

2.4 Mạng nơ-ron tích chập

2.4.1 Tổng quan

Mạng nơ-ron tích chập (Convolutional Neural Network – CNN) là một kiến trúc học sâu nổi bật, cho phép mô hình học trực tiếp từ dữ liệu đầu vào mà không cần sự can thiệp thủ công trong việc trích xuất đặc trưng. CNN đặc biệt hiệu quả trong việc nhận diện và phát hiện các mẫu trong hình ảnh, giúp phân biệt đối tượng, khuôn mặt và cảnh vật một cách chính xác. Bên cạnh đó, CNN cũng được áp dụng thành công trong nhiều lĩnh vực không liên quan đến hình ảnh, chẳng hạn như phân tích âm thanh, chuỗi thời gian và dữ liệu tín hiệu, thể hiện khả năng tổng quát hóa mạnh mẽ của kiến trúc này.

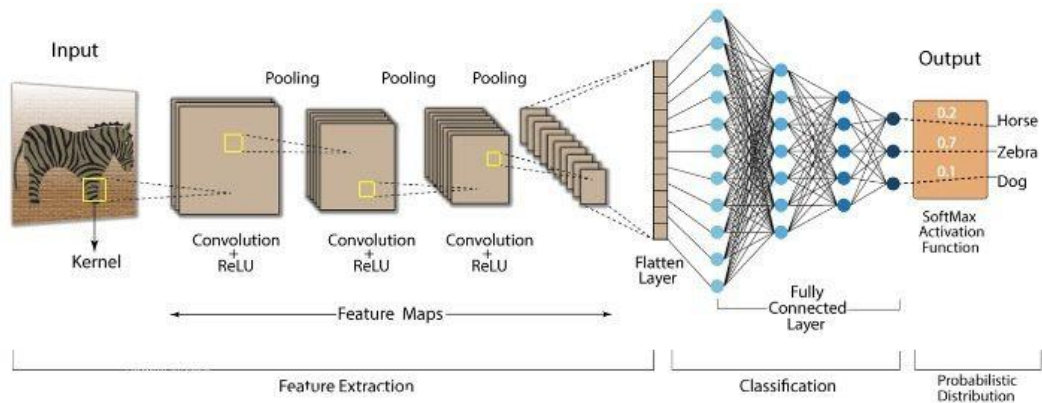
CNN có thể được xem như một dạng đặc biệt của mạng nơ-ron đa tầng (Multi-layer Neural Network), trong đó các kết nối không hoàn toàn (partial connection) được sử dụng thay vì kết nối đầy đủ như trong các mạng truyền thống.

Các mạng kết nối hoàn toàn thường dẫn đến hiện tượng quá khớp (overfitting) do số lượng lớn tham số và kết nối. Để khắc phục điều này, kỹ thuật điều chuẩn (regularization) thường được áp dụng, chẳng hạn như giảm trọng số (weight decay) hoặc bỏ kết nối ngẫu nhiên (dropout). Tuy nhiên, CNN giải quyết bài toán này theo một cách khác bằng cách tận dụng cấu trúc phân cấp trong dữ liệu, từ đó học các đặc trưng phức tạp dần lên từ những mẫu đơn giản hơn thông qua các bộ lọc. Nhờ đó, CNN có khả năng điều chuẩn tự nhiên và hiệu quả, giúp hạn chế hiện tượng quá khớp mà vẫn duy trì khả năng học sâu.

Ngày nay, CNN đóng vai trò trung tâm trong nhiều ứng dụng học sâu (Deep Learning), đặc biệt là trong lĩnh vực thị giác máy tính (Computer Vision) và các tác vụ xử lý dữ liệu tuần tự. Nhờ cơ chế chia sẻ trọng số và kết nối cục bộ, CNN làm giảm đáng kể số lượng tham số cần học, từ đó rút ngắn thời gian huấn luyện và tăng hiệu quả mô hình. Không chỉ giới hạn ở xử lý hình ảnh, CNN còn được mở rộng và áp dụng thành công trong các lĩnh vực như xử lý văn bản tự nhiên, nhận dạng giọng nói, và các bài toán liên quan đến trí tuệ nhân tạo khác. Với tốc độ phát triển nhanh chóng và khả năng thích ứng cao, CNN đã và đang trở thành một tiêu chuẩn vàng trong thiết kế mô hình học sâu hiện đại.

2.4.2 Kiến trúc

Kiến trúc truyền thống của một mạng nơ-ron tích chập (Convolutional Neural Network – CNN) bao gồm một chuỗi các tầng được sắp xếp tuần tự nhằm trích xuất và học các đặc trưng của dữ liệu đầu vào. Các tầng cơ bản trong cấu trúc này bao gồm: tầng tích chập (convolutional layer), tầng phi tuyến (activation layer, thường dùng hàm ReLU), tầng gộp (pooling layer), và tầng kết nối đầy đủ (fully connected layer), theo sau là tầng đầu ra (output layer) tương ứng với tác vụ phân loại hoặc hồi quy.



Hình 2.3 Minh họa kiến trúc mạng CNN

((Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[6])

Trong đó, các tầng tích chập và tầng gộp là những thành phần quan trọng có thể được điều chỉnh thông qua các siêu tham số (hyperparameters) như: kích thước bộ lọc (kernel size), độ trượt (stride), loại padding, và phương pháp gộp (max pooling hoặc average pooling). Việc lựa chọn và tinh chỉnh các siêu tham số này đóng vai trò then chốt trong việc kiểm soát độ sâu, khả năng trích xuất đặc trưng và hiệu suất tổng thể của mô hình CNN.

a) Tầng tích chập:

Về mặt hình học, do bộ lọc không thể vượt ra ngoài biên của ảnh, kích thước của đầu ra thường nhỏ hơn đầu vào. Với mảng đầu vào có kích thước $H \times W$, và kernel có kích thước $h \times w$, kích thước của đầu ra (nếu không sử dụng padding) sẽ là $(H - h + 1) \times (W - w + 1)$. Việc thu nhỏ kích thước này là hệ quả tất yếu của cách trượt kernel trong phạm vi hợp lệ. Tuy nhiên, để duy trì kích thước đầu ra bằng với đầu vào, người ta có thể áp dụng padding – một kỹ thuật trong đó các giá trị (thường là số 0) được đệm thêm vào các biên của ảnh đầu vào, nhằm tạo không gian đủ để kernel quét toàn bộ khu vực cần thiết.

Đầu vào		Bộ lọc		Đầu ra																	
<table><tr><td>0</td><td>1</td><td>2</td></tr><tr><td>3</td><td>4</td><td>5</td></tr><tr><td>6</td><td>7</td><td>8</td></tr></table>	0	1	2	3	4	5	6	7	8	*	<table><tr><td>0</td><td>1</td></tr><tr><td>2</td><td>3</td></tr></table>	0	1	2	3	=	<table><tr><td>19</td><td>25</td></tr><tr><td>37</td><td>43</td></tr></table>	19	25	37	43
0	1	2																			
3	4	5																			
6	7	8																			
0	1																				
2	3																				
19	25																				
37	43																				

Hình 2.4 Minh họa nhân tích chập

Tầng tích chập giúp mạng CNN học được các đặc trưng như cạnh, góc, hoa văn... ở các cấp độ khác nhau, là nền tảng cho khả năng nhận diện hình ảnh mạnh mẽ của mô hình.

b) Tầng kích hoạt (Activation Layer)

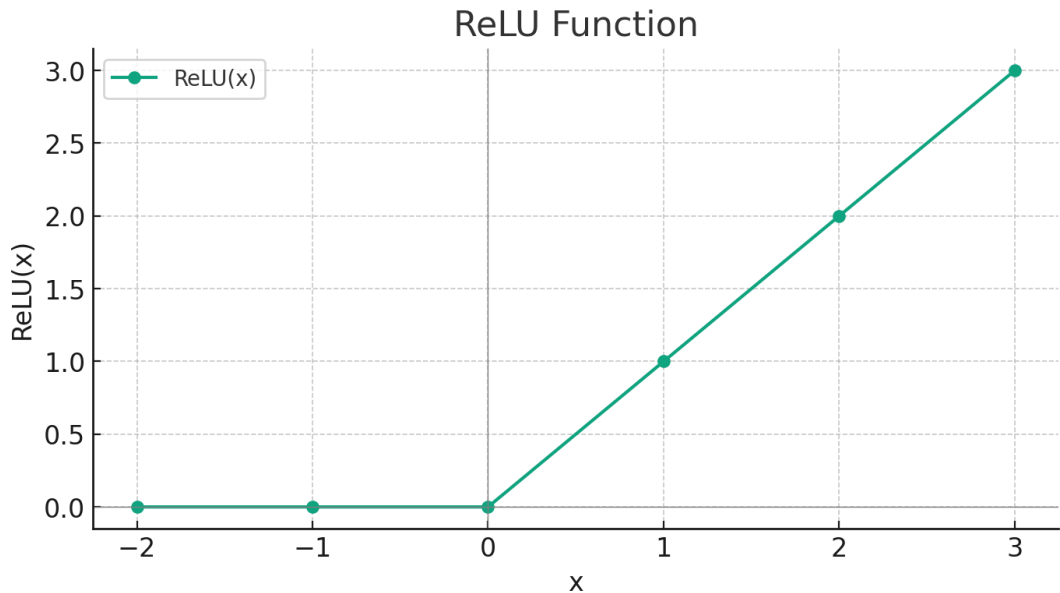
Tầng kích hoạt là một thành phần thiết yếu trong mạng nơ-ron tích chập (CNN), có vai trò giới thiệu tính phi tuyến (non-linearity) vào mô hình. Mặc dù các tầng tích chập và kết nối đầy đủ thực hiện các phép biến đổi tuyến tính lên dữ liệu đầu vào, song chính tầng kích hoạt là yếu tố then chốt giúp mạng có khả năng học và mô hình hóa các quan hệ phức tạp, phi tuyến trong dữ liệu thực tế như hình ảnh, âm thanh, hay văn bản.

Sau khi tín hiệu đi qua tầng tích chập hoặc tầng kết nối đầy đủ, nó sẽ được truyền qua một hàm kích hoạt để biến đổi đầu ra theo cách phi tuyến. Một số hàm kích hoạt phổ biến được sử dụng trong các mạng CNN bao gồm:

- ReLU (Rectified Linear Unit):

Hàm ReLU là một trong những hàm kích hoạt được sử dụng phổ biến nhất hiện nay, có công thức:

$$f(x) = \max(0, x)$$



Hình 2.5 Hàm kích hoạt ReLU

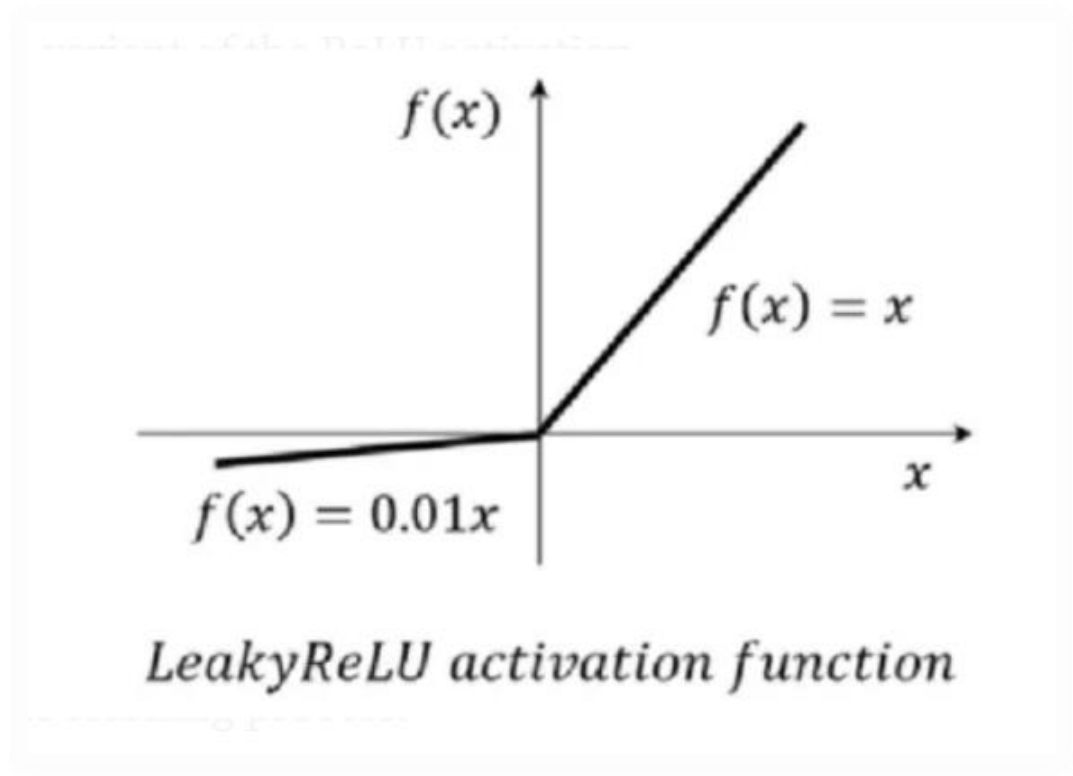
(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[7])

ReLU giúp mô hình hội tụ nhanh hơn trong quá trình huấn luyện, đồng thời làm giảm khả năng xảy ra hiện tượng biến mất đạo hàm (vanishing gradient). Tuy nhiên, một số nơ-ron có thể "chết" nếu đầu ra luôn âm, do đó dẫn đến gradient bằng 0.

- Leaky ReLU:

Để khắc phục vấn đề nơ-ron chết của ReLU, Leaky ReLU cho phép một giá trị âm nhỏ ở đầu ra:

$$f(x) = \begin{cases} x & \text{nếu } x > 0 \\ \alpha x & \text{nếu } x \leq 0 \end{cases}$$



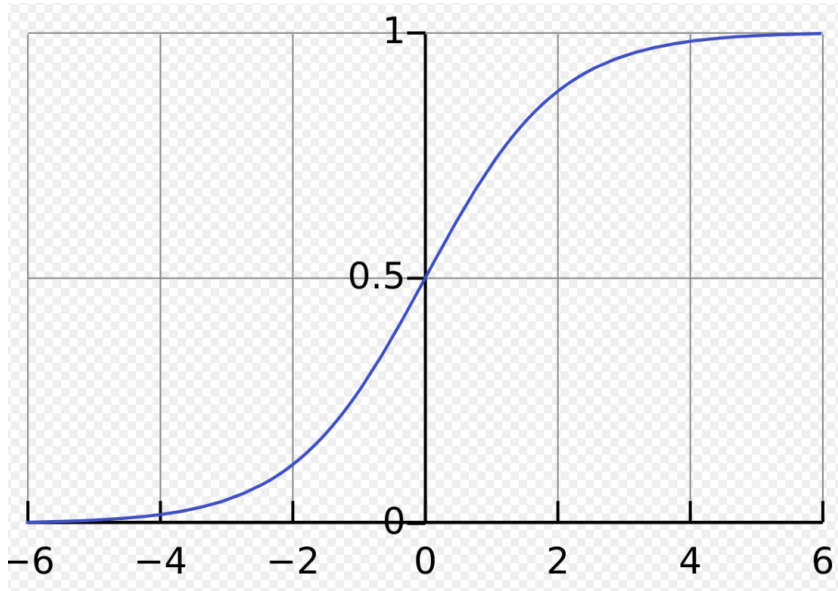
Hình 2.6 Hàm kích hoạt Leaky ReLU

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[8])

Trong đó, α là một hằng số nhỏ, thường lấy giá trị 0.01

- Sigmoid:

$$f(x) = \frac{1}{1+e^{-x}}$$



Hình 2.7 Hàm kích hoạt Sigmoid

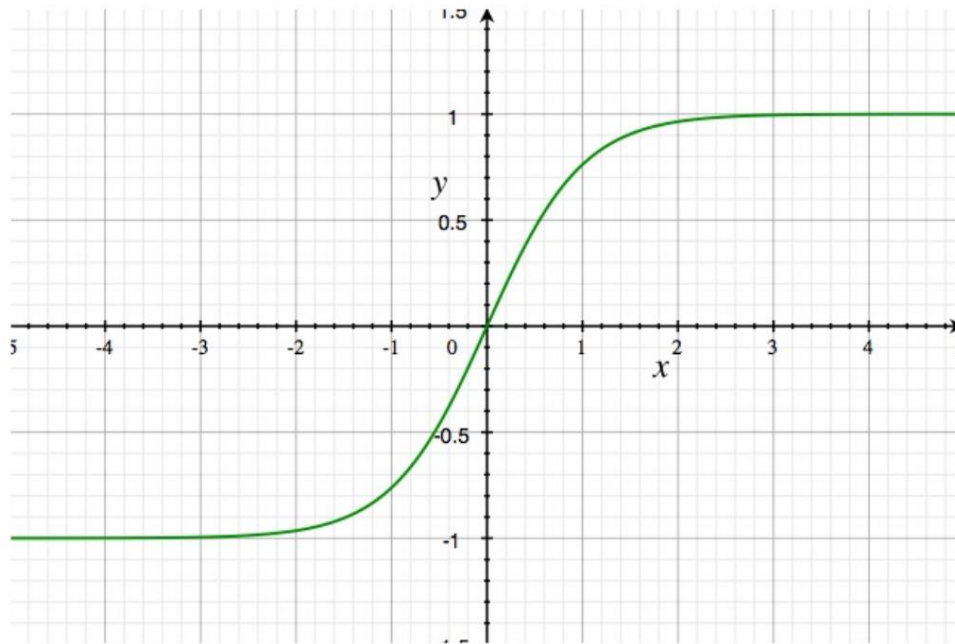
(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[9])

Hàm sigmoid nén đầu ra vào khoảng $(0,1)$, thích hợp cho các bài toán phân loại nhị phân. Tuy nhiên, sigmoid thường gây ra hiện tượng bão hòa và gradient nhỏ khi giá trị đầu vào quá lớn hoặc quá nhỏ.

- Tanh:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Hàm tanh cho đầu ra nằm trong khoảng $(-1, 1)$, giúp dữ liệu có phân phối trung tâm hơn so với sigmoid. Tuy nhiên, nó cũng gặp vấn đề về gradient nhỏ ở các đầu vào có độ lớn lớn.



Hình 2.8 Hàm kích hoạt Tanh

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[10])

Trong mạng CNN hiện đại, ReLU thường được ưu tiên sử dụng trong hầu hết các tầng nhờ tính đơn giản và hiệu quả thực nghiệm. Tầng kích hoạt thường được đặt ngay sau tầng tích chập hoặc tầng kết nối đầy đủ, đóng vai trò quyết định trong việc tăng khả năng học phi tuyến và biểu diễn đặc trưng phức tạp của mạng.

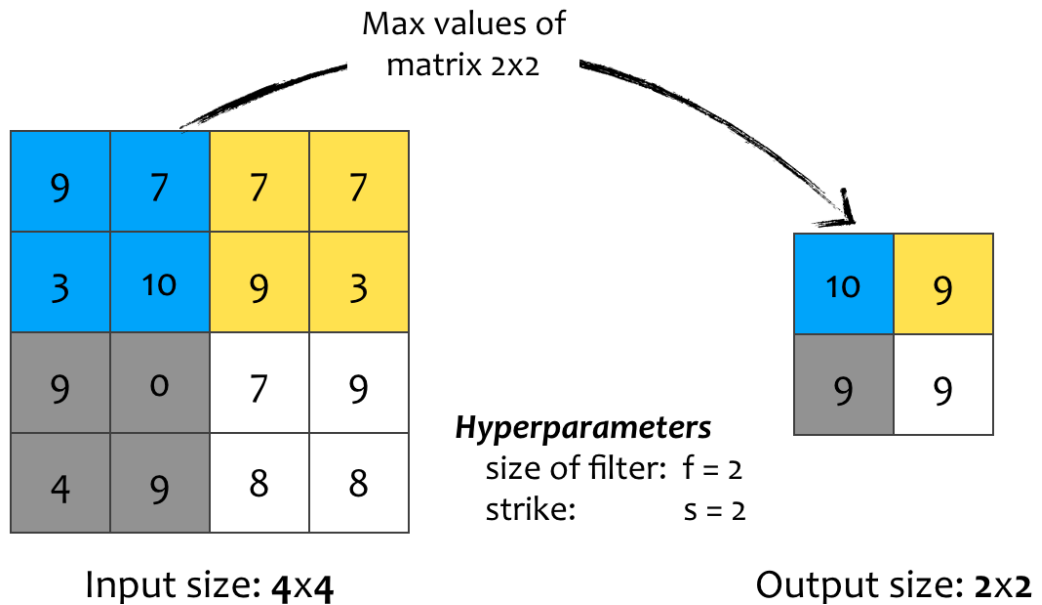
c) Tầng gộp (Pooling Layer)

Tầng gộp (Pooling Layer), hay còn gọi là tầng lấy mẫu (subsampling), là một thành phần quan trọng trong kiến trúc của mạng nơ-ron tích chập (CNN), được sử dụng sau các tầng tích chập và kích hoạt. Mục đích chính của tầng này là giảm chiều kích thước không gian (spatial dimensions) của đầu vào, từ đó giúp giảm số lượng tham số, tiết kiệm chi phí tính toán và giảm thiểu hiện tượng quá khớp (overfitting).

Tầng gộp thực hiện một phép biến đổi đơn giản nhưng hiệu quả lên từng vùng con của đầu vào (thường không chồng lấp), sử dụng một cửa sổ trượt (ví dụ 2×2 hoặc 3×3) để trích xuất đặc trưng tiêu biểu nhất trong vùng đó. Có hai loại gộp phổ biến:

- Max Pooling (Gộp cực đại):

Trong mỗi vùng con, hàm gộp cực đại giữ lại giá trị lớn nhất. Điều này giúp làm nổi bật các đặc trưng mạnh nhất trong vùng, thường được dùng phổ biến trong hầu hết các mô hình CNN hiện đại.



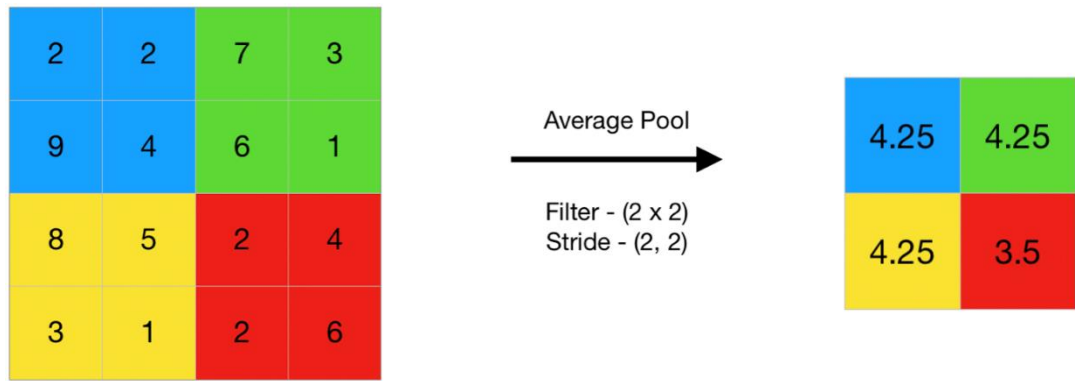
Hình 2.9 Minh họa MaxPooling

- Average Pooling (Gộp trung bình):

Thay vì chọn giá trị lớn nhất, hàm này tính trung bình cộng của các giá trị trong vùng. Average pooling thường được sử dụng trong các mô hình cũ hoặc trong giai đoạn cuối của mạng để tổng hợp thông tin.

Quá trình gộp có thể được điều chỉnh bởi các siêu tham số như:

- Kích thước cửa sổ pooling (ví dụ 2×2 , 3×3).
- Độ trượt (stride).
- Padding (đệm biên, nếu cần thiết).



Hình 2.10 Minh họa Average pooling

Việc sử dụng tầng gộp không chỉ giúp giảm kích thước dữ liệu, mà còn làm cho mạng ổn định hơn đối với các biến đổi nhỏ như dịch chuyển, quay hoặc biến dạng của đầu vào. Điều này đặc biệt hữu ích trong các tác vụ thị giác máy tính, nơi hình ảnh đầu vào có thể không được căn chỉnh hoàn hảo.

Ngoài các dạng gộp cơ bản, một số biến thể hiện đại như Global Average Pooling (gộp trung bình toàn cục) cũng được sử dụng để thay thế các tầng kết nối đầy đủ ở cuối mạng, giúp giảm tham số và cải thiện khả năng khái quát của mô hình.

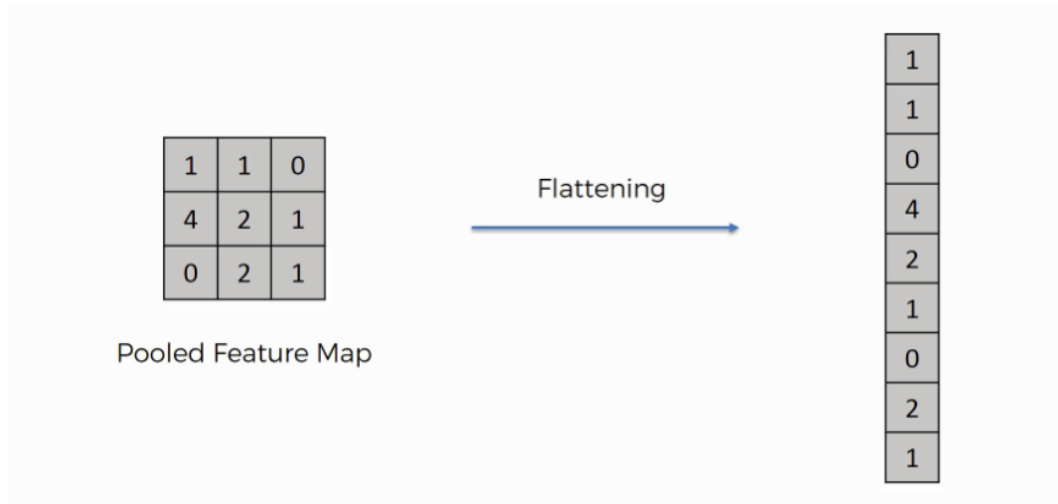
d) Tầng làm phẳng

Tầng làm phẳng (Flatten Layer) là một tầng đặc biệt trong kiến trúc mạng nơ-ron tích chập (CNN), thường được sử dụng sau các tầng tích chập và gộp, nhằm chuyển đổi đầu ra nhiều chiều (multi-dimensional) thành một vector một chiều (1D). Việc này cho phép dữ liệu có thể được đưa vào các tầng kết nối đầy đủ (Fully Connected Layer) để thực hiện các tác vụ phân loại hoặc hồi quy.

Trong các tầng tích chập, dữ liệu thường được biểu diễn dưới dạng tensor ba chiều với cấu trúc là:

$$\text{chiều cao} \times \text{chiều rộng} \times \text{số lượng kênh đặc trưng}$$

Tầng làm phẳng sẽ biến đổi tensor này thành một vector một chiều bằng cách nối tất cả các giá trị lại theo thứ tự, mà không làm mất thông tin.



Hình 2.11 Minh họa phép làm phẳng

Tầng làm phẳng không có tham số học, tức là không có trọng số hay bias, và hoạt động đơn thuần như một phép biến đổi hình dạng dữ liệu. Tuy nhiên, vai trò của nó lại rất quan trọng vì:

- Nó kết nối phần đặc trưng không gian (của ảnh) với phần phân loại của mạng.
- Là bước trung gian bắt buộc khi chuyển từ không gian 2D (ảnh) sang không gian 1D (cho đầu vào của mạng nơ-ron truyền thống hoặc tầng Dense).
- Giúp bảo toàn thông tin từ các tầng trước một cách đầy đủ trước khi đưa vào các tầng phân loại.

Như vậy, mặc dù đơn giản về mặt kỹ thuật, tầng làm phẳng là cầu nối then chốt giữa các tầng học đặc trưng và tầng ra trong một mô hình CNN hoàn chỉnh.

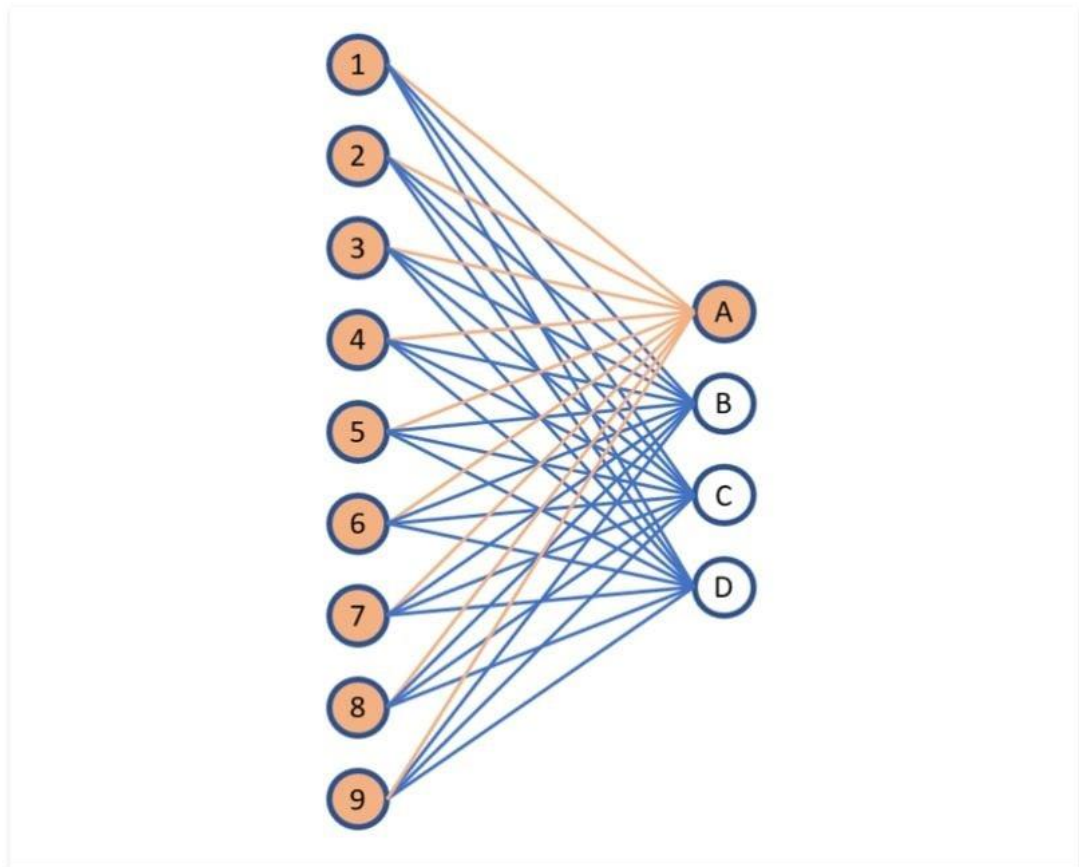
e) Tầng kết nối đầy đủ (Fully Connected Layer / Dense Layer)

Tầng kết nối đầy đủ, còn gọi là tầng Dense, là một trong những tầng cơ bản trong mạng nơ-ron sâu, đóng vai trò tổng hợp và trích xuất các đặc trưng cuối cùng từ dữ liệu đã được xử lý qua các tầng trước (tầng tích chập, tầng gộp, tầng làm phẳng).

Ở tầng này, mỗi nơ-ron được kết nối với tất cả các nơ-ron của tầng trước đó, tạo nên mạng lưới kết nối dày đặc (fully connected). Nhờ cấu trúc này, tầng kết nối đầy đủ có khả năng học được các mối quan hệ phức tạp giữa các đặc trưng, từ đó giúp mạng thực hiện tốt các nhiệm vụ phân loại hoặc hồi quy.

Công thức tính toán tại một nơ-ron trong tầng kết nối đầy đủ được biểu diễn như sau:

$$y = f\left(\sum_i w_i x_i + b\right)$$



Hình 2.12 Minh họa tầng kết nối đầy đủ

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[11])

Trong đó:

- x_i là các đầu vào từ tầng trước,
- w_i là trọng số tương ứng,

- là hệ số bias,
- f là hàm kích hoạt (thường là ReLU, sigmoid, tanh, ...).

Tầng Fully Connected thường được đặt gần cuối mạng CNN, sau khi các đặc trưng đã được làm phẳng và trích xuất đầy đủ. Tuy nhiên, nhược điểm của tầng này là số lượng tham số rất lớn, dẫn đến chi phí tính toán cao và dễ gây quá khớp nếu không được điều chỉnh tốt.

f) Tầng đầu ra (Output Layer)

Tầng đầu ra là tầng cuối cùng trong mạng nơ-ron, có nhiệm vụ chuyển đổi đặc trưng đã học được thành kết quả dự đoán phù hợp với bài toán cụ thể như phân loại hoặc hồi quy.

Cấu trúc và hàm kích hoạt của tầng đầu ra phụ thuộc vào loại nhiệm vụ:

- Phân loại nhị phân:

Tầng đầu ra thường có một nơ-ron duy nhất với hàm kích hoạt sigmoid, giúp dự đoán xác suất thuộc về một lớp nhất định. Giá trị đầu ra nằm trong khoảng $(0, 1)$.

- Phân loại đa lớp:

Tầng đầu ra có số nơ-ron bằng số lớp cần phân loại, sử dụng hàm kích hoạt softmax để chuyển đổi đầu ra thành xác suất thuộc mỗi lớp. Softmax đảm bảo tổng các xác suất bằng 1, thuận tiện cho việc phân loại đa lớp.

- Hồi quy:

Tầng đầu ra có thể có một hoặc nhiều nơ-ron tùy thuộc vào số chiều của biến đầu ra, thường không sử dụng hàm kích hoạt hoặc sử dụng hàm tuyến tính để dự đoán giá trị liên tục.

Tầng đầu ra quyết định cách mạng phản hồi và đưa ra dự đoán cuối cùng, là bước then chốt để đánh giá hiệu quả của toàn bộ mô hình.

2.5 Mạng nơ-ron ResNet50

2.5.1 Giới thiệu chung về ResNet50

ResNet50 (Residual Network, 50 layers) là một kiến trúc mạng nơ-ron tích chập sâu (CNN) được đề xuất bởi nhóm nghiên cứu của Microsoft trong bài báo "Deep Residual Learning for Image Recognition" tại hội nghị CVPR 2016. Mục tiêu chính của ResNet là giải quyết vấn đề mất mát thông tin và độ chính xác suy giảm khi mạng càng sâu, thông qua cơ chế Residual Learning (học phần dư).

2.5.2 Ý tưởng chính: Residual Learning

Các mạng sâu thông thường gặp khó khăn trong quá trình huấn luyện vì hiện tượng gradient biến mất và degradation problem (hiệu suất giảm khi mạng sâu hơn). ResNet giải quyết vấn đề này bằng cách sử dụng kết nối tắt (skip connections) – cho phép tín hiệu đi "tắt" qua một số lớp mà không bị biến đổi.

Residual block (khối phần dư) thực hiện:

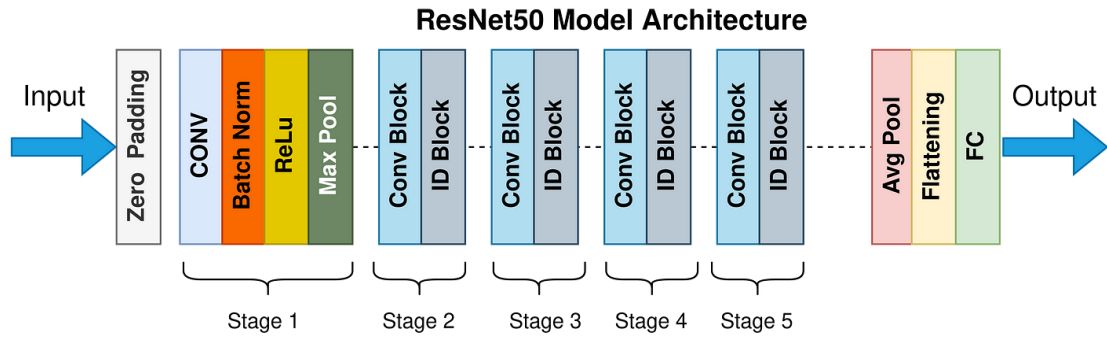
$$y = F(x, W) + x$$

Trong đó:

- x : đầu vào
- $F(x, W)$: hàm học được (thường là 2 hoặc 3 lớp tích chập)
- x được cộng trực tiếp vào đầu ra để tạo thành đầu ra cuối cùng y

2.5.3 Kiến trúc tổng thể của ResNet50

ResNet50 gồm 50 lớp có trọng số, được chia thành các khối residual như sau:



Hình 2.13 Mô hình ResNet50

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[12])

Bảng 2.1 Bảng mô tả kiến trúc mạng ResNet50

Giai đoạn	Cấu trúc	Số lần lặp	Đầu ra
Conv1	7x7 Conv, 64, stride 2 + MaxPool	1	112×112×64
Conv2_x	1x1, 64 → 3x3, 64 → 1x1, 256	3	56×56×256
Conv3_x	1x1, 128 → 3x3, 128 → 1x1, 512	4	28×28×512
Conv4_x	1x1, 256 → 3x3, 256 → 1x1, 1024	6	14×14×1024
Conv5_x	1x1, 512 → 3x3, 512 → 1x1, 2048	3	7×7×2048
Cuối cùng	Average Pooling + Fully Connected	1	1000 lớp Softmax

Tổng cộng: 49 lớp conv + 1 fully connected = 50 lớp có trọng số

2.5.4 Bottleneck block trong ResNet50

Khác với ResNet18/34 (dùng 2 lớp conv), ResNet50 dùng Bottleneck block gồm 3 lớp:

- 1x1 Conv (giảm chiều)
- 3x3 Conv

- 1x1 Conv (tăng chiều trở lại)

Cấu trúc này giúp:

- Giảm số lượng tham số
- Giảm tính toán
- Dễ huấn luyện mạng sâu

2.5.5 Ưu điểm của ResNet50

a) Khả năng huấn luyện mạng sâu

- ResNet giải quyết vấn đề degradation (độ chính xác giảm khi tăng số lớp) bằng cơ chế skip connection, giúp tín hiệu gradient lan truyền tốt trong quá trình backpropagation.
- Nhờ đó, ResNet50 (với 50 lớp) vẫn có thể huấn luyện hiệu quả và đạt độ chính xác cao hơn nhiều mạng CNN truyền thống cùng độ sâu.

b) Skip connection giúp tránh mất thông tin

- Cơ chế residual learning cho phép truyền thẳng thông tin đầu vào qua các lớp, giúp bảo toàn đặc trưng gốc và tránh mất mát thông tin trong quá trình học.

c) Khả năng tái sử dụng đặc trưng

- Các khối residual giúp mạng học tốt hơn các biến thể tinh tế của đặc trưng, thay vì học lại toàn bộ đầu vào mới, nhờ đó tối ưu hóa quá trình học và cải thiện khả năng tổng quát.

d) Hiệu quả tính toán với bottleneck

- Kiến trúc Bottleneck block ($1 \times 1 \rightarrow 3 \times 3 \rightarrow 1 \times 1$) giảm số lượng tham số và FLOPs so với kiến trúc thường, trong khi vẫn giữ được khả năng học biểu diễn sâu.

e) Tính linh hoạt và dễ mở rộng

- Có thể dễ dàng mở rộng lên các phiên bản sâu hơn như ResNet101, ResNet152 hoặc kết hợp vào các mô hình phức tạp như Mask R-CNN, FPN, UNet++, v.v.

f) Hiệu quả đã được kiểm chứng

- ResNet50 đã đạt kết quả rất tốt trong nhiều bài toán thị giác máy tính: ImageNet classification, object detection, segmentation...

2.5.6 Nhược điểm của ResNet50

a) Kích thước mô hình lớn

- ResNet50 có khoảng 25 triệu tham số, khá nặng so với các mô hình nhẹ như MobileNet, EfficientNet, gây khó khăn khi triển khai trên các thiết bị tài nguyên hạn chế (như điện thoại, IoT).

b) Yêu cầu tài nguyên tính toán cao

- Do số lớp nhiều và cấu trúc phức tạp, ResNet50 đòi hỏi GPU mạnh và bộ nhớ lớn khi huấn luyện hoặc suy luận trên tập dữ liệu lớn.

c) Không phải luôn tốt hơn mạng nông

- Trong một số bài toán đơn giản, mạng sâu như ResNet50 có thể dễ bị overfitting và kém hiệu quả hơn mạng nhẹ nếu không có đủ dữ liệu hoặc không điều chỉnh đúng hyperparameters.

d) Không tận dụng tốt thông tin không gian cục bộ

- Mặc dù rất tốt về học đặc trưng toàn cục, nhưng ResNet50 không có cơ chế chuyên biệt để học quan hệ không gian phức tạp như Attention hoặc Transformer, nên có thể hạn chế trong một số bài toán yêu cầu chi tiết không gian (ví dụ: segmentation cao cấp).

e) Thiết kế thủ công, ít linh hoạt

- Kiến trúc ResNet được thiết kế bằng tay (hand-crafted) nên không tối ưu về mặt tự động hóa như các mô hình tìm kiếm kiến trúc (NAS – Neural Architecture Search).

2.6 Mạng nơ-ron DenseNet 121

2.6.1 Người giới thiệu và thời gian

a) Người đề xuất:

DenseNet được giới thiệu bởi nhóm nghiên cứu gồm Gao Huang, Zhuang Liu, Laurens van der Maaten, và Kilian Q. Weinberger.

b) Thời gian công bố:

DenseNet lần đầu tiên được trình bày trong bài báo "Densely Connected Convolutional Networks" tại hội nghị CVPR (Conference on Computer Vision and Pattern Recognition) năm 2017.

c) Link bài báo gốc:

"Densely Connected Convolutional Networks" (CVPR 2017)[4]

2.6.2 Ý tưởng chính của DenseNet

- DenseNet đưa ra ý tưởng cốt lõi là kết nối dày đặc (dense connectivity) giữa các lớp (layers) trong mạng nơ-ron sâu.
- Thay vì truyền dữ liệu chỉ qua từng lớp liên tiếp như các mạng truyền thống (như VGG, ResNet), DenseNet kết nối mỗi lớp với tất cả các lớp trước đó theo kiểu trực tiếp.
- Cụ thể:
 - Với một mạng có L layers, mỗi layer sẽ nhận đầu vào từ tất cả các layer trước đó (gồm layer 0 đến layer $l-1$).
 - Nếu bạn đặt x_0 là đầu vào ban đầu, và x_l là đầu ra của lớp l , thì:

$$x_l = H_l([x_0, x_1, \dots, x_{l-1}])$$

(trong đó là phép nối kênh (concatenation) của tất cả các feature maps trước đó, và h_l là một tổ hợp các hàm như Batch Normalization, ReLU và Convolution).

- Ý nghĩa:
 - Giảm hiện tượng mất mát thông tin (vanishing-gradient) nhờ liên kết ngắn mạch.

- Khuyến khích tái sử dụng đặc trưng (feature reuse) — các feature hữu ích được chia sẻ suốt toàn bộ mạng, làm cho việc học hiệu quả hơn.
- Cấu trúc nhỏ gọn — số lượng tham số giảm đáng kể so với mạng truyền thống có độ sâu tương đương.

2.6.3 Số lượng tham số

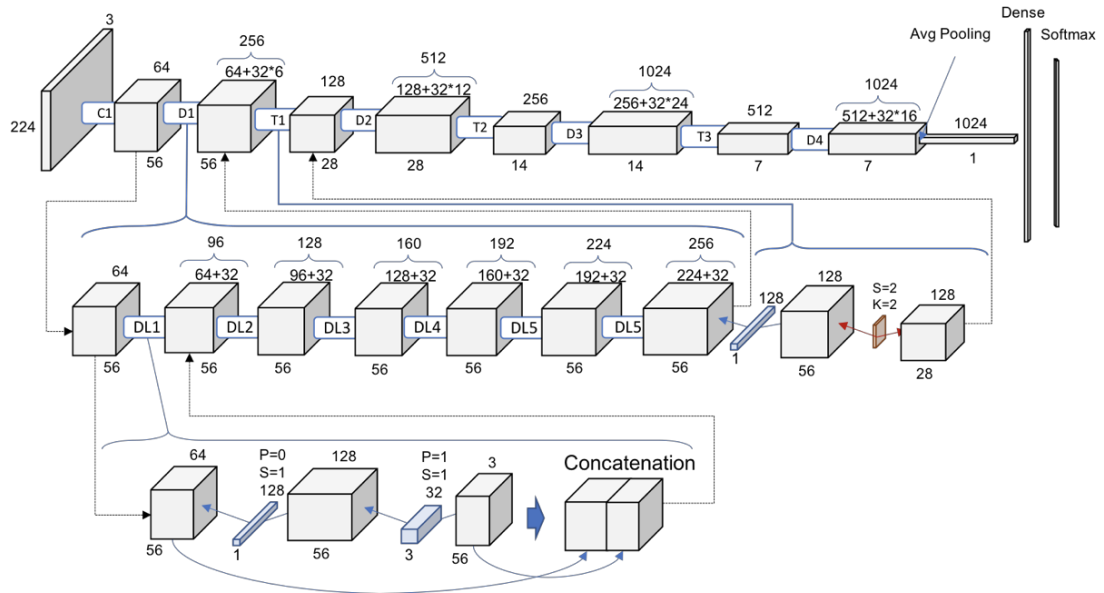
- Số lượng tham số trong DenseNet nhỏ hơn đáng kể so với các mạng có độ sâu tương đương như ResNet.
- Cách tính số tham số:
 - Mỗi layer chỉ tạo thêm một số lượng nhỏ feature maps mới, thường được gọi là growth rate (thường ký hiệu là k).
 - Ví dụ: Nếu growth rate $k = 32$ thì mỗi lớp thêm 32 feature maps mới, còn feature maps cũ thì chỉ được copy (không thêm tham số).

a) Cụ thể:

- Một DenseNet-121 (121 layers) chỉ có khoảng 8 triệu tham số.
- Trong khi một ResNet-152 (152 layers) có khoảng 60 triệu tham số.

2.6.4 Kiến trúc DenseNet-121

DenseNet-121 được chia làm các phần:



Hình 2.14 Mô hình DenseNet-121

(Nguồn: Hình ảnh được trích dẫn từ tài liệu tham khảo[13])

Bảng 2.2 Bảng mô tả các thành phần ở mô hình DenseNet-121

Giai đoạn	Các lớp thành phần	Chi tiết
0	Conv Layer	7×7 Convolution, stride 2 64 filters Theo sau là BatchNorm + ReLU
1	Pooling Layer	3×3 Max Pooling, stride 2
2	Dense Block 1	6 Bottleneck layers
3	Transition Layer 1	1×1 Convolution 2×2 Average Pooling, stride 2
4	Dense Block 2	12 Bottleneck layers
5	Transition Layer 2	1×1 Convolution

6	Dense Block 3	24 Bottleneck layers
7	Transition Layer 3	1×1 Convolution 2×2 Average Pooling, stride 2
8	Dense Block 4	16 Bottleneck layers
9	Classification Layer	Global Average Pooling (over feature maps) Fully Connected Layer (số lượng units = số lớp output, như 1000 cho ImageNet) Softmax

2.6.5 Ưu điểm của DenseNet

a) Tăng cường truyền gradient (Improved Gradient Flow):

- Do mỗi lớp kết nối trực tiếp tới tất cả các lớp trước đó, gradient được truyền ngược dễ dàng hơn.
- Giảm mạnh vấn đề vanishing gradient trong mạng sâu.
- Tái sử dụng đặc trưng (Feature Reuse):
- Các đặc trưng (feature maps) được sử dụng lại nhiều lần qua các layers sau.
- Giúp mạng học được nhiều khái niệm đa cấp độ với ít tham số hơn.

b) Số lượng tham số ít:

- Mặc dù DenseNet rất sâu, nhưng nhờ việc chỉ sinh ra một lượng nhỏ feature mới mỗi lớp (growth rate thấp), tổng tham số vẫn thấp.
- Ví dụ: DenseNet-121 có khoảng 8M tham số, ít hơn rất nhiều so với ResNet-152 (60M tham số).

c) Tăng hiệu quả học tập:

- DenseNet cần ít epoch hơn để đạt độ chính xác tốt so với các mạng cùng độ sâu như ResNet.
- Tốc độ hội tụ khi training cũng nhanh hơn.

d) Giảm overfitting trên datasets nhỏ:

- Do số lượng tham số thấp và khả năng sử dụng lại thông tin tốt, DenseNet thường ít overfit hơn trên các tập dữ liệu không lớn.

2.6.6 Nhược điểm của DenseNet**a) Tốn bộ nhớ (Memory Intensive):**

- Vì mỗi layer phải lưu trữ feature maps của tất cả layers trước đó, nên lượng RAM/GPU memory cần thiết cao hơn so với mạng truyền thống.
- Rất dễ hết bộ nhớ khi training DenseNet trên hình ảnh kích thước lớn hoặc batch size lớn.

b) Chi phí tính toán cao (Computational Cost):

- Mỗi layer phải xử lý đầu vào ngày càng lớn do việc nối (concatenation) feature maps.
- Dẫn đến số lượng phép nhân tích chập (FLOPs) khá lớn dù số tham số nhỏ.

c) Thiết kế phức tạp hơn:

- Do cấu trúc nối đặc biệt, DenseNet khó implement thủ công hơn các mạng như VGG hay ResNet.
- Cần tối ưu kỹ các hàm memory-efficient để training mô hình lớn.

d) Khó mở rộng cho bài toán rất lớn:

- Trong các ứng dụng cần hàng ngàn layers (ví dụ siêu phân giải ảnh - super-resolution, hoặc xử lý video), DenseNet gốc khó mở rộng do chi phí RAM tăng quá nhanh.

2.7 Tóm tắt chương

Chương 2 trình bày các kỹ thuật và mô hình phổ biến được sử dụng để giải quyết bài toán nhận dạng ảnh u não. Trước tiên, chương đưa ra định hướng tiếp

cận bài toán theo hướng kết hợp giữa kỹ thuật truyền thống và các mô hình học sâu. Sau đó, một số phương pháp cụ thể được giới thiệu như K-Means Clustering – một thuật toán phân cụm không giám sát đơn giản nhưng hiệu quả, với phần trình bày đầy đủ về định nghĩa, cách hoạt động, công thức tính toán, ưu và nhược điểm.

Tiếp theo, chương tập trung vào các mô hình mạng nơ-ron tích chập (CNN) – nền tảng quan trọng trong xử lý ảnh y tế. Các mô hình CNN hiện đại như ResNet50 và DenseNet121 được trình bày chi tiết, từ ý tưởng chủ đạo (Residual Learning, kết nối dày đặc giữa các lớp), kiến trúc tổng thể, đến phân tích ưu và nhược điểm của từng mô hình. Những đặc điểm này giúp làm rõ tiềm năng và tính ứng dụng cao của các kiến trúc mạng nơ-ron sâu trong nhận dạng hình ảnh y học, qua đó làm cơ sở cho việc lựa chọn và xây dựng mô hình phù hợp ở các chương tiếp theo.

CHƯƠNG 3: THỰC NGHIỆM

3.1 Môi trường thực nghiệm

3.1.1 Phần cứng

- Cấu hình thiết bị được sử dụng trong quá trình thực nghiệm như sau:
- CPU: AMD Ryzen 5 5600H (6 nhân / 12 luồng, xung nhịp tối đa 4.2 GHz)
- RAM: 16.0 GB DDR4
- GPU: NVIDIA GeForce GTX 1650 4GB GDDR6 (phiên bản laptop, TDP 50W)
- Ổ cứng: SSD NVMe 512GB
- Ngoài ra tôi còn huấn luyện trên Colab là môi trường phần cứng do Google giới thiệu và phát triển.

3.1.2 Phần mềm

Các phần mềm và thư viện được sử dụng trong quá trình xây dựng, huấn luyện và thử nghiệm mô hình bao gồm:

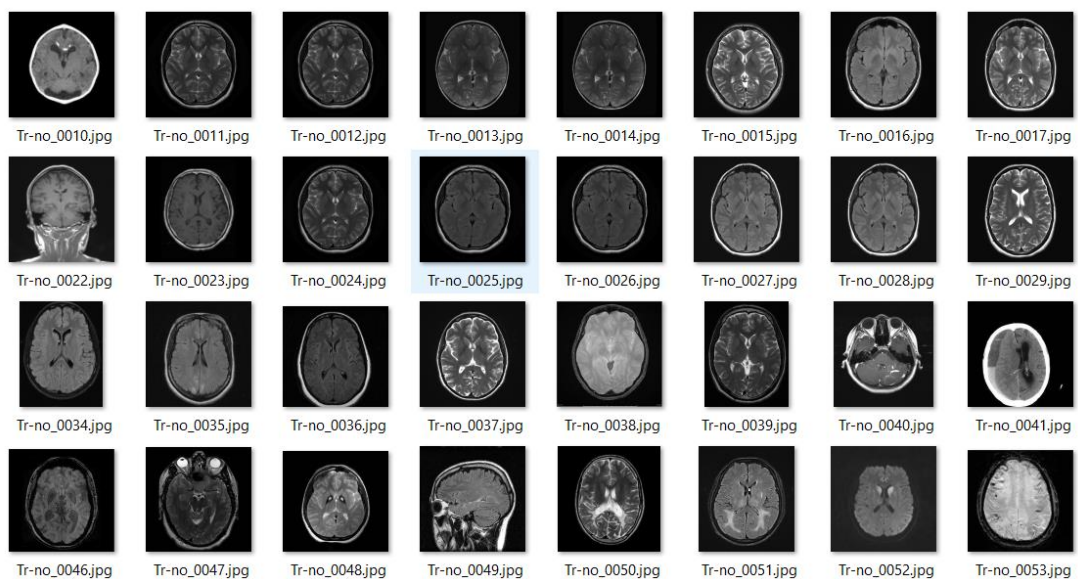
- ❖ Hệ điều hành: Windows 10 Home, 64-bit
- ❖ Ngôn ngữ lập trình: Python 3.12
- ❖ Các thư viện
 - tensorflow: Dùng mô hình tiền huấn luyện DenseNet121, Dùng trong transfer learning, Mô hình ResNet50 tiền huấn luyện, One-hot encoding nhãn cho bài toán phân loại
 - matplotlib.pyplot: Vẽ đồ thị, hình ảnh, trực quan hóa kết quả mô hình
 - sklearn.utils.shuffle: Trộn ngẫu nhiên dữ liệu đầu vào
 - os: Quản lý đường dẫn, thư mục, file hệ thống
 - cv2(OpenCV): Xử lý ảnh: đọc, chuyển đổi định dạng, resize, v.v.
 - numpy: Tính toán ma trận, vector, mảng số học hiệu suất cao
- ❖ Môi trường phát triển: PyCharm

3.2 Dữ liệu thực nghiệm

Quá trình thực nghiệm được tối tiến hành trên bộ dữ liệu Brain Tumor MRI Dataset từ Kaggle, bao gồm các ảnh chụp cộng hưởng từ MRI của não người. Mỗi ảnh là ảnh chụp MRI(Magnetic Resonance Imaging) ở dạng 2D với độ phân giải trung bình và định dạng chuẩn(thường là .jpg hoặc png)

3.2.1 Nội dung

Bộ dữ liệu chứa khoảng 7023 ảnh, bao gồm hơn 5000 ảnh cho tập train và hơn 1000 ảnh cho tập test, được chia đều hoặc gần đều giữa các lớp.



Hình 3.1 Một số hình ảnh trong bộ dữ liệu

Ảnh được chia thành 4 loại tương ứng với 4 nhãn (classes):

- glioma (u thần kinh đệm)
- meningioma (u màng não)
- pituitary tumor (u tuyến yên)
- no tumor (không có khối u)

Bảng 3.1 Bảng thông tin bộ dữ liệu

Thông tin	Chi tiết
-----------	----------

Tổng số hình ảnh	7023
Số lượng lớp phân loại	4
Số ảnh trong tập train	5712
Số ảnh trong tập test	1311

3.2.2 Cấu trúc thư mục

Dữ liệu được tổ chức thành các thư mục tương ứng với từng loại u não, thuận tiện cho việc trích xuất và tiền xử lý.

Brain_Tumor_Dataset/

Training/

glioma/

meningioma/

notumor/

pituitary/

Testing/

glioma/

meningioma/

notumor/

pituitary/

Test/

- Mỗi thư mục con chứa các ảnh .jpg hoặc .jpeg tương ứng.

3.3 Tiền xử lý dữ liệu

Ảnh MRI trong bộ dữ liệu có nhiều khác biệt về độ sáng, tương phản và nhiễu sinh lý. Một số ảnh có độ tương phản thấp hoặc nhiễu nền cao, đòi hỏi bước tiền xử lý như tăng cường tương phản, lọc nhiễu hoặc chuẩn hóa cường độ. Bộ dữ liệu này có xu hướng khá cân bằng giữa các lớp, mặc dù vẫn có sự chênh lệch nhỏ

về số lượng ảnh từng loại. Điều này giúp hạn chế tình trạng mô hình thiên lệch nhưng vẫn cần kiểm tra kỹ nếu áp dụng các kỹ thuật học máy nhạy với mất cân bằng.

Trước khi huấn luyện mô hình, ảnh sẽ được tiền xử lý như: chuyển đổi kích thước, chuẩn hóa pixel, và có thể áp dụng phép biến đổi ảnh để tăng độ đa dạng dữ liệu.

```
# =====
# 2. Cấu hình Image Generator
# =====
img_size = 224
batch_size = 64

train_datagen = ImageDataGenerator(
    rescale=1./255,
    rotation_range=10,
    zoom_range=0.1,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True
)

test_datagen = ImageDataGenerator(rescale=1./255)
```

Hình 3.2 Các tham số của ImageDataGenerator

Ý nghĩa từng tham số

- `rescale=1./255`: Ảnh thường có giá trị pixel từ 0 đến 255 chia cho 255 để đưa toàn bộ ảnh về khoảng $[0,1]$ giúp mô hình học nhanh hơn, giảm gradient vanishing/exploding.
- `rotation_range=10`: Xoay ảnh ngẫu nhiên trong khoảng ± 10 độ. Giúp mô hình không bị phụ thuộc vào hướng cố định của ảnh đầu vào.
- `zoom_range=0.1`: Phóng to hoặc thu nhỏ ảnh trong khoảng $\pm 10\%$. Mô phỏng trường hợp bệnh lý có thể to nhỏ khác nhau.

- `width_shift_range=0.1` và `height_shift_range=0.1` Dịch ảnh theo chiều ngang và dọc tối đa 10% kích thước ảnh. Hữu ích nếu vùng u não không cố định tại một vị trí.
- `horizontal_flip=True` Lật ảnh theo chiều ngang giúp tăng độ đa dạng trong ảnh đầu vào.

```
train_generator = train_datagen.flow_from_directory(
    train_dir,
    target_size=(img_size, img_size),
    batch_size=batch_size,
    class_mode='categorical'
)

test_generator = test_datagen.flow_from_directory(
    test_dir,
    target_size=(img_size, img_size),
    batch_size=batch_size,
    class_mode='categorical',
    shuffle=False
)
```

Hình 3.3 Tạo Generator từ thư mục ảnh

- `flow_from_directory()`: Đọc ảnh từ thư mục có cấu trúc và Tự động gán nhãn theo tên thư mục con (0, 1, 2, 3). Chuyển đổi thành các cặp (x, y): x là ảnh, y là nhãn one-hot vector.
- `target_size=(img_size, img_size)`: Resize toàn bộ ảnh về kích thước 224x224 (vì DenseNet yêu cầu đầu vào cố định). Nếu ảnh gốc có tỷ lệ khác thì sẽ bị kéo dãn (biến dạng nhẹ), có thể cải thiện thêm bằng padding giữ nguyên tỷ lệ (nếu muốn).
- `batch_size=64`: Số lượng ảnh được đưa vào một lần huấn luyện (mini-batch).
- `shuffle=False`: Giữ nguyên thứ tự ảnh test để dễ phân tích kết quả sau này (ví dụ như hiển thị ảnh và dự đoán).

3.4 Huấn luyện với mô hình DenseNet 121

Đầu tiên, ta cấu hình đường dẫn đến bộ dữ liệu

```
# =====
# 1. Cấu hình đường dẫn
# =====
data_dir = "/content/drive/My Drive/DATN"
train_dir = os.path.join(data_dir, "Training")
test_dir = os.path.join(data_dir, "Testing")
```

Hình 3.4 Định nghĩa thư mục dữ liệu

- data_dir chứa toàn bộ dữ liệu.
- train_dir, test_dir là thư mục con chứa ảnh huấn luyện và kiểm thử.

```
# Load mô hình DenseNet121 với trọng số ImageNet
base_model = DenseNet121(weights="imagenet", include_top=False, input_shape=(224, 224, 3))
base_model.trainable = False # Đóng băng mô hình gốc

# Thêm các lớp tùy chỉnh
x = GlobalAveragePooling2D()(base_model.output)
x = Dense(units=512, activation="relu")(x)
x = Dropout(0.3)(x)
output = Dense(units=4, activation="softmax")(x) # 4 lớp đầu ra

# Tạo mô hình hoàn chỉnh
model = Model(inputs=base_model.input, outputs=output)
```

Hình 3.5 Xây dựng mô hình

- weights="imagenet": sử dụng trọng số đã được huấn luyện trên ImageNet (tập dữ liệu lớn).
- include_top=False: bỏ phần lớp Fully Connected gốc, chỉ lấy feature extractor.
- trainable=False: đóng băng DenseNet121, không cập nhật trọng số trong quá trình training.
- GlobalAveragePooling2D: thay flatten bằng việc lấy trung bình mỗi feature map.
- Dense(512, activation="relu"): lớp fully connected 512 neuron.

- Dropout(0.3): tránh overfitting, random bỏ 30% neuron khi training.
- Dense(4, activation="softmax"): lớp đầu ra với 4 lớp (corresponding to 4 types of tumors).
- Kết nối từ đầu vào DenseNet121 đến đầu ra custom.

```
# Biên dịch mô hình
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy"])
```

Hình 3.6 Biên dịch mô hình

- optimizer="adam": thuật toán tối ưu hóa.
- loss="categorical_crossentropy": dùng cho bài toán phân loại nhiều lớp.
- metrics=["accuracy"]: theo dõi độ chính xác.

```
# =====
# 5. Callbacks
# =====
earlystop = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
checkpoint = ModelCheckpoint(model_path, monitor='val_accuracy', save_best_only=True)
```

Hình 3.7 Thiết lập callbacks

- EarlyStopping: dừng huấn luyện nếu 5 epoch liên tiếp không cải thiện val_loss, và tự động khôi phục weights tốt nhất.
- ModelCheckpoint: lưu file mô hình mỗi khi val_accuracy đạt mức cao mới.

```
# =====
# 6. Huấn luyện
# =====
history = model.fit(
    train_generator,
    validation_data=test_generator,
    epochs=100,
    callbacks=[earlystop, checkpoint]
)
```

Hình 3.8 Huấn luyện mô hình

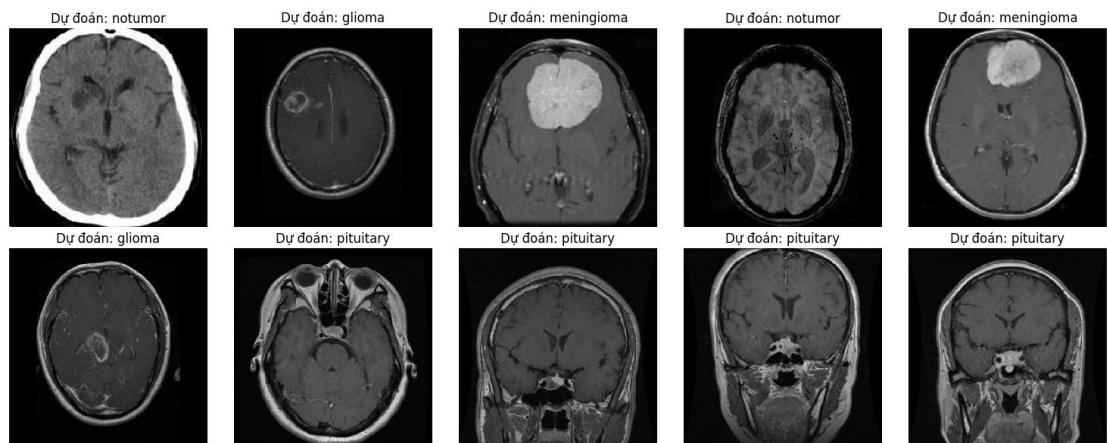
- Huấn luyện tối đa 100 epoch, nhưng thường dừng sớm nhờ EarlyStopping.
- Lưu lại lịch sử `history.history` để vẽ đồ thị sau.

```

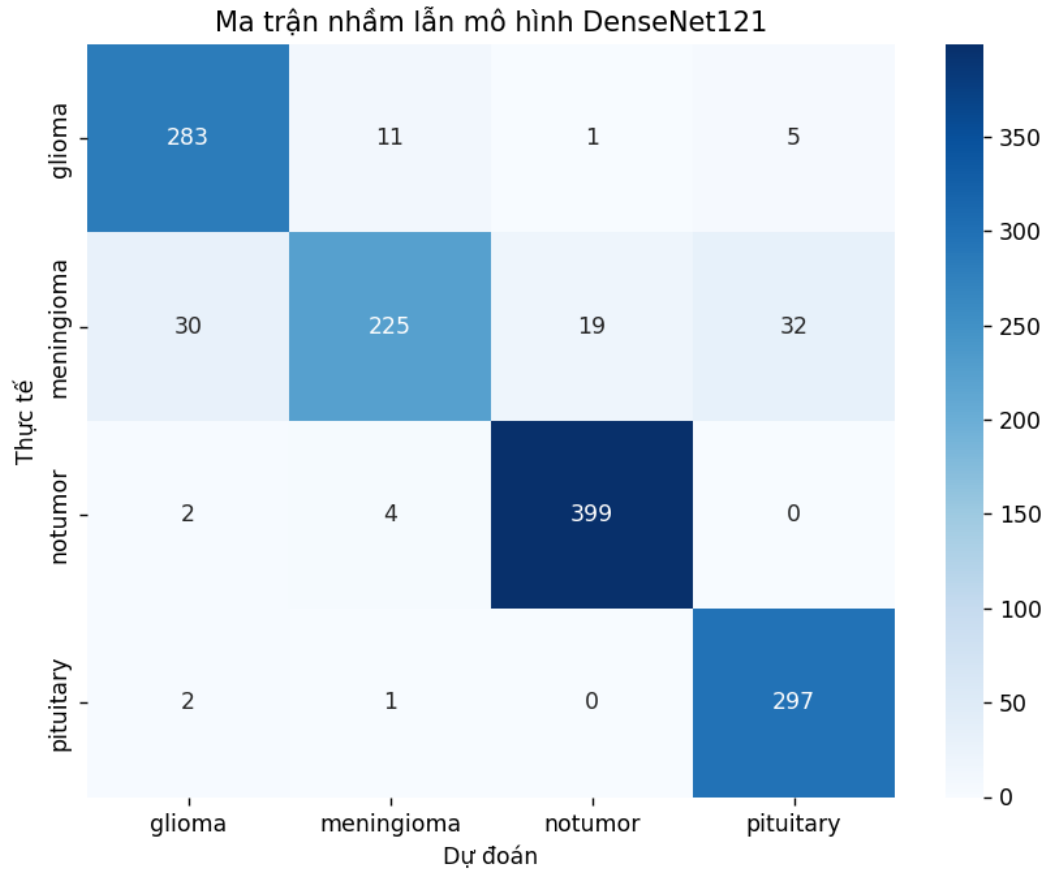
Epoch 9/50
90/90 ----- 98s 1s/step - accuracy: 0.9038 - loss: 0.2660 - val_accuracy: 0.8757 - val_loss: 0.3359
Epoch 10/50
90/90 ----- 141s 1s/step - accuracy: 0.9069 - loss: 0.2724 - val_accuracy: 0.8528 - val_loss: 0.3618
Epoch 11/50
90/90 ----- 98s 1s/step - accuracy: 0.9143 - loss: 0.2591 - val_accuracy: 0.8696 - val_loss: 0.3308
Epoch 12/50
90/90 ----- 97s 1s/step - accuracy: 0.9081 - loss: 0.2473 - val_accuracy: 0.8833 - val_loss: 0.2994
Epoch 13/50
90/90 ----- 97s 1s/step - accuracy: 0.9122 - loss: 0.2399 - val_accuracy: 0.8749 - val_loss: 0.3257
Epoch 14/50
90/90 ----- 101s 1s/step - accuracy: 0.9148 - loss: 0.2477 - val_accuracy: 0.8970 - val_loss: 0.2698
Epoch 15/50
90/90 ----- 96s 1s/step - accuracy: 0.9208 - loss: 0.2270 - val_accuracy: 0.8909 - val_loss: 0.2878
Epoch 16/50
90/90 ----- 96s 1s/step - accuracy: 0.9202 - loss: 0.2209 - val_accuracy: 0.8863 - val_loss: 0.2829
Epoch 17/50
90/90 ----- 142s 1s/step - accuracy: 0.9237 - loss: 0.2168 - val_accuracy: 0.8856 - val_loss: 0.2758
Epoch 18/50
90/90 ----- 143s 1s/step - accuracy: 0.9293 - loss: 0.2089 - val_accuracy: 0.8802 - val_loss: 0.3005
Epoch 19/50
90/90 ----- 96s 1s/step - accuracy: 0.9232 - loss: 0.2080 - val_accuracy: 0.9047 - val_loss: 0.2466
Epoch 20/50
90/90 ----- 142s 1s/step - accuracy: 0.9366 - loss: 0.1850 - val_accuracy: 0.8978 - val_loss: 0.2544
Epoch 21/50
90/90 ----- 97s 1s/step - accuracy: 0.9260 - loss: 0.2068 - val_accuracy: 0.8970 - val_loss: 0.2543
Epoch 22/50
90/90 ----- 96s 1s/step - accuracy: 0.9358 - loss: 0.1879 - val_accuracy: 0.8917 - val_loss: 0.2638
Epoch 23/50
90/90 ----- 100s 1s/step - accuracy: 0.9353 - loss: 0.1779 - val_accuracy: 0.8833 - val_loss: 0.2758
Epoch 24/50
90/90 ----- 96s 1s/step - accuracy: 0.9340 - loss: 0.1876 - val_accuracy: 0.8924 - val_loss: 0.2658

```

Hình 3.9 Quá trình huấn luyện mô hình DenseNet121



Hình 3.10 Dự đoán với 10 hình ảnh trong tập test



Hình 3.11 Ma trận nhầm lẫn của mô hình

3.5 Huấn luyện với mô hình ResNet50

Mô hình ResNet50 được sử dụng cho bài toán phân loại u não gồm 4 lớp: glioma, meningioma, notumor và pituitary. Mô hình được tải với trọng số đã huấn luyện sẵn từ ImageNet và bỏ đi phần fully-connected gốc (`include_top=False`). Tất cả các lớp của ResNet50 được đóng băng (không cập nhật trọng số trong quá trình huấn luyện) để tận dụng khả năng trích xuất đặc trưng mạnh mẽ từ ảnh.

Dữ liệu huấn luyện và kiểm tra được tổ chức theo cấu trúc thư mục và được nạp thông qua `ImageDataGenerator`, trong đó tập huấn luyện được áp dụng kỹ thuật tăng cường dữ liệu (augmentation) như xoay, phóng to, dịch chuyển và lật ảnh để giảm overfitting. Ảnh được resize về kích thước 224x224 và chuẩn hóa về khoảng $[0, 1]$.

```

# Tải mô hình ResNet50 (không gồm top layer)
base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Đồng bộ các layer gốc
for layer in base_model.layers:
    layer.trainable = False

# Thêm các tầng phân loại mới
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(128, activation='relu')(x)
predictions = Dense(4, activation='softmax')(x) # 4 lớp: glioma, meningioma, notumor, pituitary

# Tạo model hoàn chỉnh
model = Model(inputs=base_model.input, outputs=predictions)

# Compile model
model.compile(optimizer=Adam(learning_rate=0.0001),
              loss='categorical_crossentropy',
              metrics=['accuracy'])

# Callbacks
checkpoint = ModelCheckpoint("resnet50_brain_tumor.h5", save_best_only=True, monitor='val_accuracy')
earlystop = EarlyStopping(patience=5, restore_best_weights=True)

```

Hình 3.12 Cấu trúc mô hình ResNet50

Trên đầu mô hình gốc, các lớp phân loại mới được thêm vào gồm GlobalAveragePooling2D, Dense(128, relu) và Dense(4, softmax) để đưa ra dự đoán xác suất cho 4 lớp. Mô hình được biên dịch với optimizer Adam (learning rate = 0.0001), hàm mất mát categorical_crossentropy và metric accuracy.

Trong quá trình huấn luyện, hai callback được sử dụng:

- ModelCheckpoint: lưu lại mô hình có độ chính xác validation cao nhất.
- EarlyStopping: dừng sớm nếu độ chính xác trên tập validation không cải thiện sau 5 epoch, giúp tránh overfitting.

Mô hình được huấn luyện trong 100 epoch, và mô hình tốt nhất được lưu dưới tên resnet50_brain_tumor.h5.

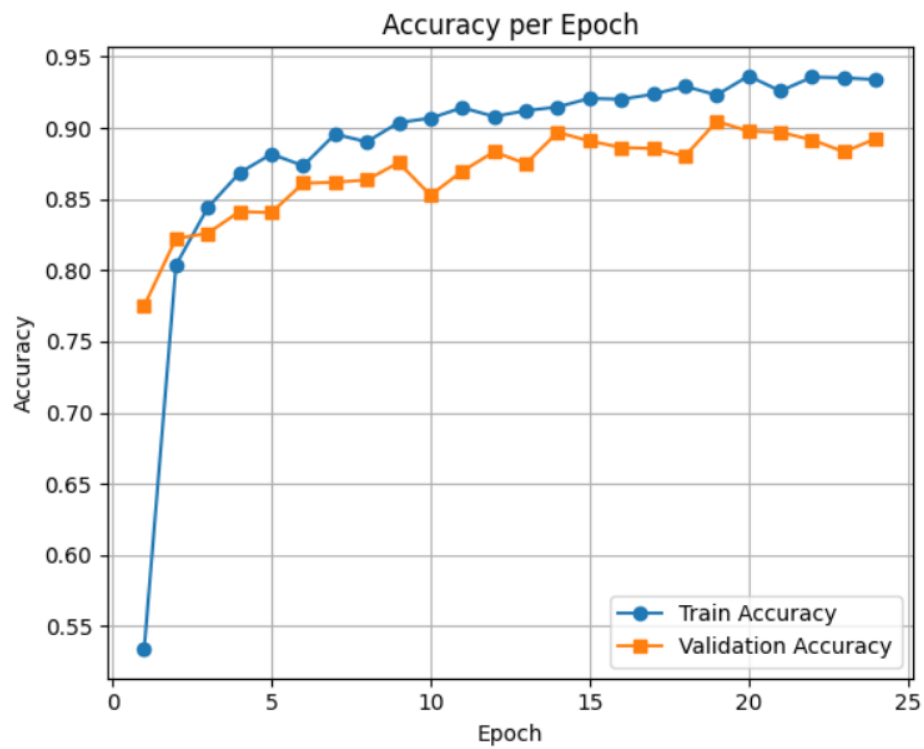
Epoch 12/100	90/90	98s	1s/step	- accuracy: 0.9144 - loss: 0.2396 - val_accuracy: 0.8886 - val_loss: 0.3384
Epoch 13/100	90/90	139s	1s/step	- accuracy: 0.9214 - loss: 0.2334 - val_accuracy: 0.8589 - val_loss: 0.3384
Epoch 14/100	90/90	99s	1s/step	- accuracy: 0.9228 - loss: 0.2344 - val_accuracy: 0.8787 - val_loss: 0.3083
Epoch 15/100	90/90	98s	1s/step	- accuracy: 0.9178 - loss: 0.2303 - val_accuracy: 0.8947 - val_loss: 0.2643
Epoch 16/100	90/90	96s	1s/step	- accuracy: 0.9257 - loss: 0.2187 - val_accuracy: 0.8818 - val_loss: 0.2831
Epoch 17/100	90/90	96s	1s/step	- accuracy: 0.9195 - loss: 0.2284 - val_accuracy: 0.8711 - val_loss: 0.3148
Epoch 18/100	90/90	98s	1s/step	- accuracy: 0.9283 - loss: 0.2114 - val_accuracy: 0.8986 - val_loss: 0.2444
Epoch 19/100	90/90	97s	1s/step	- accuracy: 0.9322 - loss: 0.2038 - val_accuracy: 0.8863 - val_loss: 0.2716
Epoch 20/100	90/90	97s	1s/step	- accuracy: 0.9351 - loss: 0.1939 - val_accuracy: 0.8924 - val_loss: 0.2746
Epoch 21/100	90/90	98s	1s/step	- accuracy: 0.9304 - loss: 0.1942 - val_accuracy: 0.9001 - val_loss: 0.2450
Epoch 22/100	90/90	141s	1s/step	- accuracy: 0.9313 - loss: 0.1869 - val_accuracy: 0.8986 - val_loss: 0.2453
Epoch 23/100	90/90	98s	1s/step	- accuracy: 0.9302 - loss: 0.1962 - val_accuracy: 0.9077 - val_loss: 0.2332
Epoch 24/100	90/90	98s	1s/step	- accuracy: 0.9364 - loss: 0.1894 - val_accuracy: 0.9092 - val_loss: 0.2366
Epoch 25/100	90/90	98s	1s/step	- accuracy: 0.9301 - loss: 0.1894 - val_accuracy: 0.9039 - val_loss: 0.2499
Epoch 26/100	90/90	98s	1s/step	- accuracy: 0.9379 - loss: 0.1750 - val_accuracy: 0.9191 - val_loss: 0.2151
Epoch 27/100	90/90	98s	1s/step	- accuracy: 0.9430 - loss: 0.1696 - val_accuracy: 0.9047 - val_loss: 0.2320
Epoch 28/100	90/90	102s	1s/step	- accuracy: 0.9392 - loss: 0.1714 - val_accuracy: 0.9100 - val_loss: 0.2213

Hình 3.13 Quá trình huấn luyện mô hình ResNet50

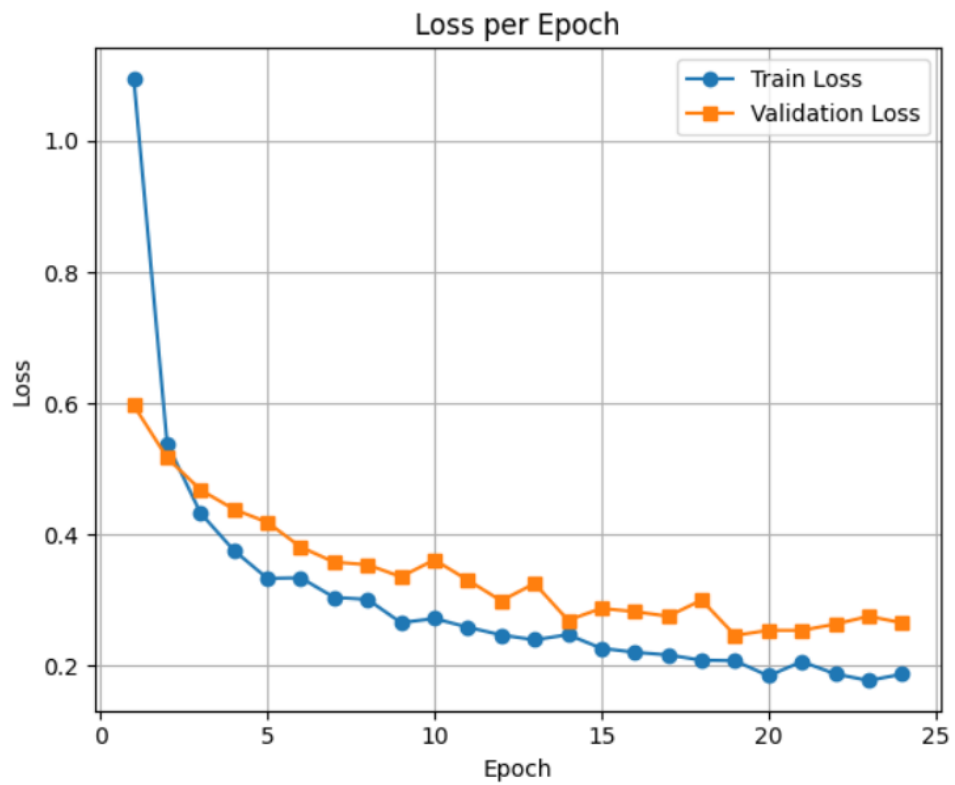
3.6 Các kết quả thực nghiệm

3.6.1 Mô hình DenseNet121

Mô hình được huấn luyện với tối đa 100 epoch, tuy nhiên quá trình huấn luyện đã dừng sớm tại epoch thứ 24 do độ chính xác trên tập validation không cải thiện sau 5 epoch liên tiếp. Kết quả đạt được là độ chính xác 93,4% trên tập huấn luyện và 90,47% trên tập kiểm tra.



Hình 3.14 Biểu đồ Accuracy của DenseNet121



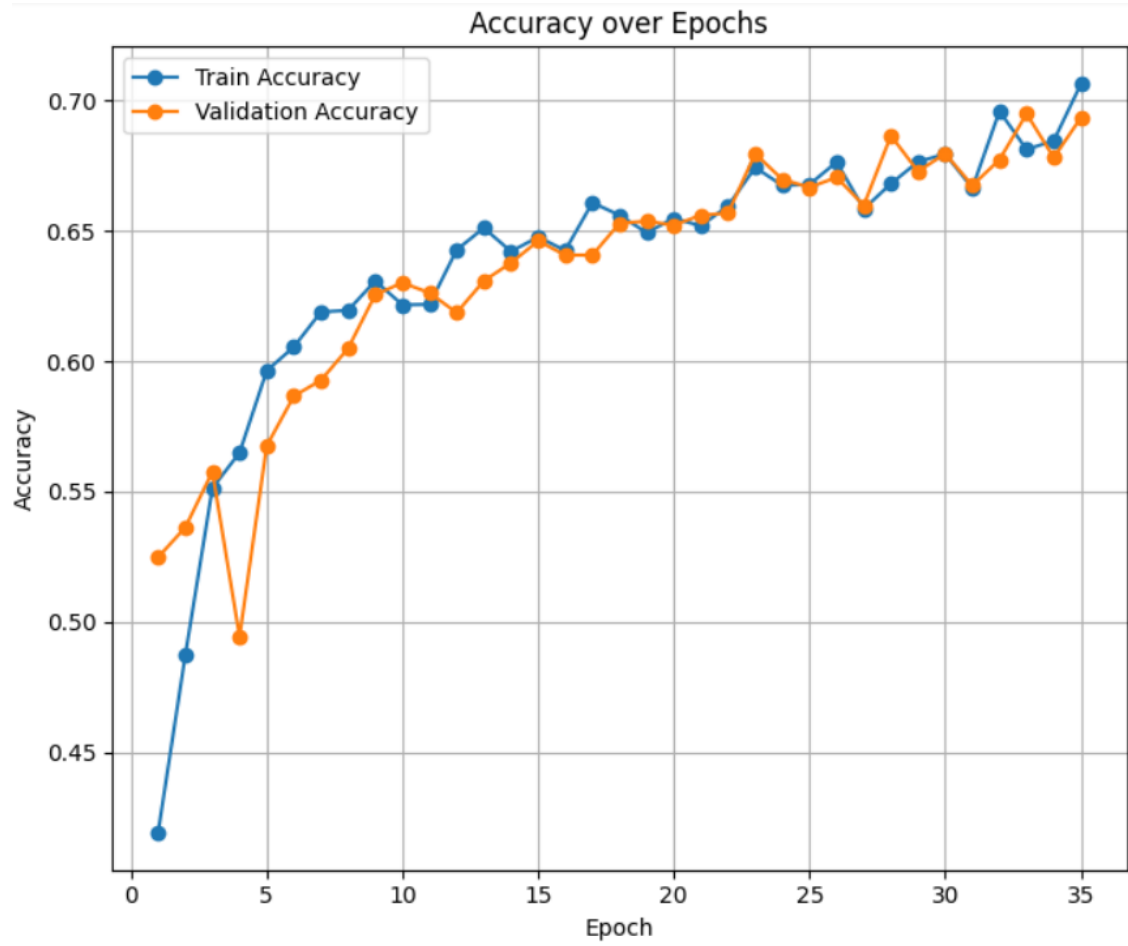
Hình 3.15 Biểu đồ Loss của DenseNet121

Trong suốt quá trình huấn luyện 24 epoch, độ chính xác trên tập huấn luyện (Train Accuracy) đã tăng trưởng đều đặn từ 53.36% lên đến hơn 93.40%, cho thấy mô hình học hiệu quả từ dữ liệu. Đồng thời, độ chính xác trên tập kiểm định (Validation Accuracy) cũng được cải thiện đáng kể, từ mức ban đầu 77.50% lên đến khoảng 90.47% tại epoch 19, rồi dao động ổn định quanh mức này. Sự chênh lệch nhỏ giữa hai giá trị độ chính xác phản ánh rằng mô hình học khá tốt mà không bị overfitting nghiêm trọng, đồng thời vẫn duy trì được khả năng tổng quát hóa tốt trên dữ liệu chưa từng thấy.

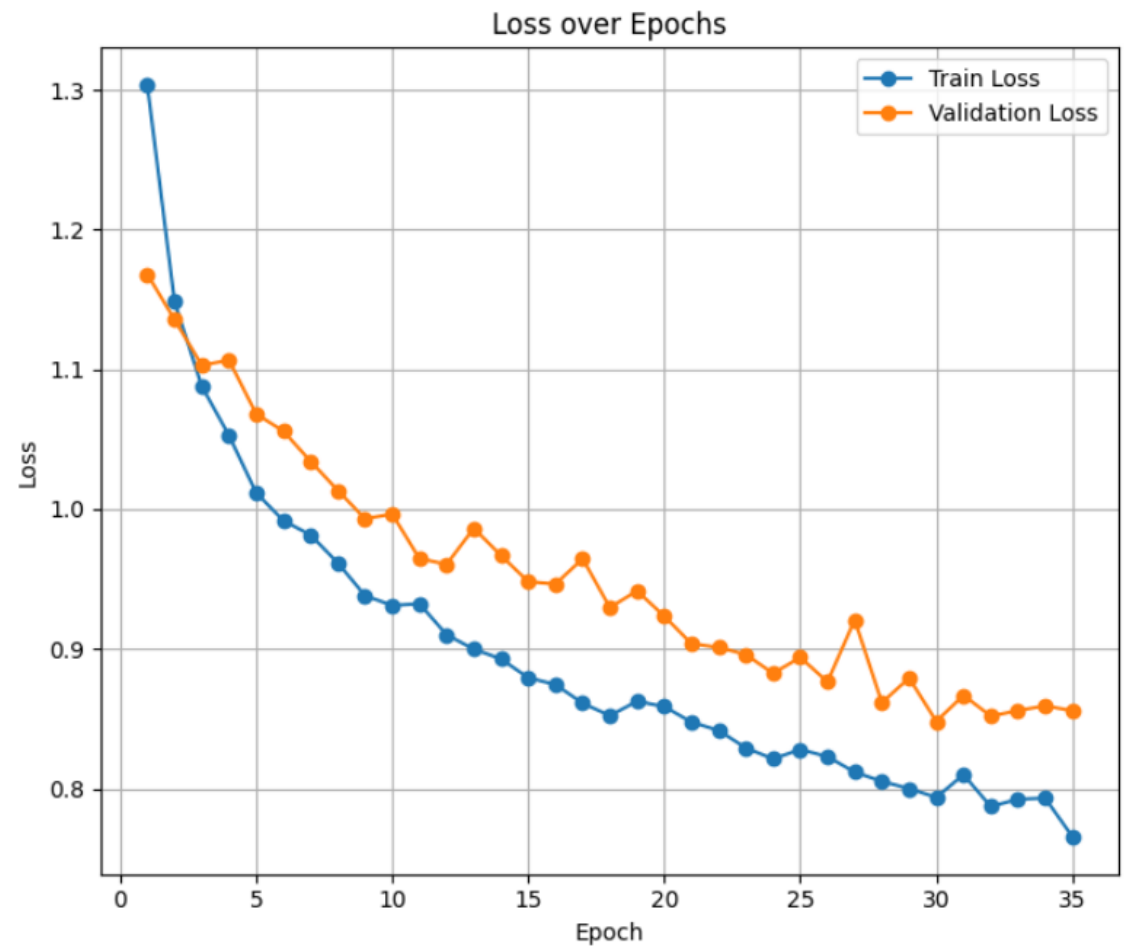
Mất mát trên tập huấn luyện (Train Loss) đã giảm đáng kể từ 1.0953 xuống còn khoảng 0.1876 sau 24 epoch, cho thấy quá trình tối ưu hóa diễn ra hiệu quả. Mất mát trên tập kiểm định (Validation Loss) cũng có xu hướng giảm đều, từ 0.5972 ban đầu xuống mức thấp nhất là 0.2466 tại epoch 19, trước khi dao động nhẹ trong các epoch sau. Sự đồng bộ giữa hai đường loss cho thấy mô hình không chỉ học tốt mà còn duy trì được độ ổn định khi áp dụng lên dữ liệu kiểm định, một dấu hiệu tích cực cho khả năng triển khai thực tế.

3.6.2 Mô hình ResNet50

Mô hình được huấn luyện với tối đa 100 epoch, tuy nhiên quá trình huấn luyện đã dừng sớm tại epoch thứ 35 do độ chính xác trên tập validation không cải thiện sau 5 epoch liên tiếp. Kết quả đạt được là độ chính xác 70,63% trên tập huấn luyện và 67,96% trên tập kiểm tra.



Hình 3.16 Biểu đồ Accuracy của ResNet50



Hình 3.17 Biểu đồ Loss của ResNet50

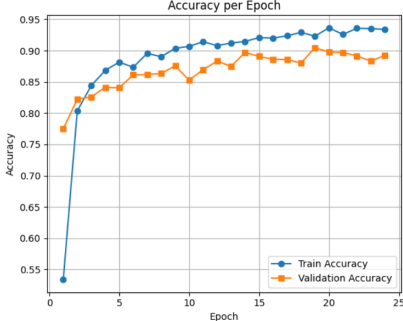
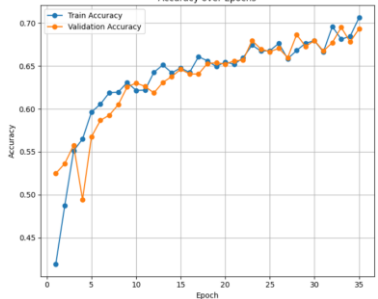
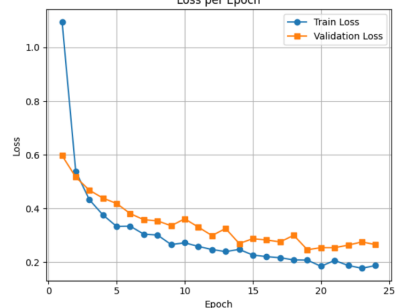
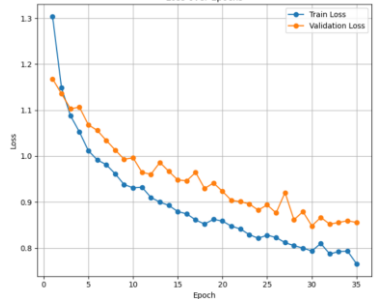
- Train Accuracy và Validation Accuracy

Trong quá trình huấn luyện, độ chính xác trên tập huấn luyện (train accuracy) đã được cải thiện rõ rệt từ 41.92% ở epoch đầu tiên lên đến hơn 70.63% tại epoch 35. Điều này cho thấy mô hình học dần được đặc trưng từ dữ liệu huấn luyện. Độ chính xác trên tập kiểm định (validation accuracy) cũng tăng đều, từ 52.48% ở epoch 1 lên 69.49% ở epoch 33 và duy trì ổn định quanh mức này. Tuy vậy, độ chính xác trên tập test (67.96%) thấp hơn một chút so với validation, điều này có thể cho thấy mô hình vẫn có thể được cải thiện thêm về khả năng tổng quát hóa.

- Train Loss và Validation Loss

Mất mát (loss) trên tập huấn luyện giảm dần từ 1.3033 xuống còn 0.7657 sau 35 epoch, chứng minh quá trình tối ưu hóa đang tiến triển đúng hướng. Trong khi đó, mất mát trên tập kiểm định (validation loss) giảm từ 1.1680 xuống khoảng 0.8557 ở epoch cuối, và dao động quanh mức ổn định từ epoch 20 trở đi. Dù validation loss giảm chậm hơn so với train loss, nhưng không có dấu hiệu rõ ràng của overfitting. Điều này cho thấy mô hình đang học khá ổn định, tuy nhiên có thể cần tinh chỉnh thêm (như learning rate hoặc thêm regularization) để tối ưu hiệu suất cao hơn.

3.6.3 So sánh kết quả 2 mô hình

Mô hình	DenseNet121	ResNet50
Độ chính xác(Accuracy)		
Hàm mất mát(Loss)		

Hình 3.18 So sánh giữa hai mô hình

- Về độ chính xác:

Trong quá trình huấn luyện, mô hình DenseNet121 thể hiện khả năng học hiệu quả hơn so với ResNet50. Cụ thể, DenseNet121 nhanh chóng đạt độ chính xác huấn luyện trên 90% chỉ sau khoảng 15 epoch, trong khi ResNet50 cần nhiều

epoch hơn để tiệm cận ngưỡng này. Mức độ hội tụ nhanh hơn cho thấy kiến trúc DenseNet giúp mô hình tiếp nhận và lan truyền thông tin tốt hơn giữa các lớp, nhờ vào cơ chế kết nối dày đặc.

Khi đánh giá trên tập kiểm tra (test set), DenseNet121 tiếp tục khẳng định ưu thế với độ chính xác 90.47%, vượt trội so với 67.96% của ResNet50. Đây là một chỉ số quan trọng cho thấy DenseNet121 không chỉ học tốt trên dữ liệu huấn luyện mà còn tổng quát hóa tốt hơn trên dữ liệu mới. Điều này gợi ý rằng DenseNet121 phù hợp hơn với bài toán phân loại trong bối cảnh hiện tại.

- Về hàm mất mát

Hàm mất mát trong huấn luyện là chỉ số phản ánh mức độ sai lệch giữa dự đoán của mô hình và nhãn thực tế. Trong các epoch đầu, cả hai mô hình đều có xu hướng giảm loss khá đều. Tuy nhiên, DenseNet121 nhanh chóng đạt loss dưới 0.3, trong khi ResNet50 dao động quanh mức 0.8–1.0 trong giai đoạn đầu và chỉ giảm dần về 0.76. Điều này cho thấy DenseNet121 học đặc trưng hiệu quả hơn, ít mắc lỗi hơn trong quá trình huấn luyện.

Xét trên tập validation, DenseNet121 tiếp tục thể hiện ưu thế với giá trị loss thấp nhất là 0.2466, so với khoảng 0.8481 của ResNet50 tại thời điểm tốt nhất. Điều này không chỉ minh chứng cho khả năng học sâu tốt hơn của DenseNet121 mà còn phản ánh độ ổn định và khả năng tránh hiện tượng overfitting tốt hơn. Loss thấp và độ chính xác cao là dấu hiệu điển hình cho một mô hình học sâu hiệu quả và đáng tin cậy.

3.7 Tóm tắt chương

Chương 3 trình bày quá trình thực nghiệm nhằm đánh giá hiệu quả của các mô hình học sâu trong bài toán nhận dạng ảnh u não. Mở đầu là mô tả về môi trường thực nghiệm, bao gồm phần cứng, phần mềm, và các thư viện sử dụng. Phần tiếp theo giới thiệu bộ dữ liệu được sử dụng trong nghiên cứu, cùng với cấu trúc thư mục và các thông số kỹ thuật liên quan đến tiền xử lý và định dạng dữ liệu.

Sau đó, hai mô hình chính – DenseNet121 và ResNet50 – được triển khai và huấn luyện trên cùng một tập dữ liệu, dưới các điều kiện giống nhau để đảm bảo tính công bằng trong so sánh. Cuối cùng, chương trình bày và phân tích kết quả thực nghiệm thông qua các chỉ số đánh giá như độ chính xác (accuracy và hàm mất mát (loss).

Kết quả thực nghiệm cho thấy DenseNet121 vượt trội hơn so với ResNet50 trên cả phương diện độ chính xác và khả năng tổng quát hóa mô hình. Với thiết kế liên kết chặt chẽ giữa các lớp (dense connections), DenseNet121 giúp giảm thiểu hiện tượng mất thông tin và cải thiện việc lan truyền gradient trong mạng sâu. Nhờ đó, mô hình học được nhiều đặc trưng hơn với ít tham số hơn, dẫn đến hiệu suất tốt hơn trong nhận dạng ảnh u não. Đây là cơ sở để đề xuất DenseNet121 là lựa chọn ưu tiên cho bài toán trong phạm vi nghiên cứu này.

CHƯƠNG 4: XÂY DỰNG SẢN PHẨM

4.1 Flask

4.1.1 Giới thiệu về Flask

Flask là một Web Framework rất nhẹ của Python, dễ dàng giúp người mới bắt đầu học Python có thể tạo ra website nhỏ. Flask cũng dễ mở rộng để xây dựng các ứng dụng web phức tạp. Điểm nổi bật khi sử dụng Flask để lập trình web là ta sẽ rất ít bị phụ thuộc bên thứ 3, do đó đề phòng được các lỗi bảo mật.



Hình 4.1 Flask

4.1.2 Các tính năng của Flask Framework:

- Phát triển máy chủ
- Phát triển trình gỡ lỗi
- Khả năng tương thích công cụ dựa trên ứng dụng Google
- Nhiều tiện ích mở rộng cho các tính năng mong muốn
- Hỗ trợ bảo mật cookie
- Cung cấp xử lý HTTP request
- API độc đáo và mạch lạc, hỗ trợ RESTful API

- Dễ dàng triển khai

Với mục tiêu xây dựng một Backend server không quá phức tạp nhưng phải xử lý được các yêu cầu của người dùng với tốc độ nhanh, tôi quyết định nghiên cứu và sử dụng framework Flask.

4.2 Bootstrap

4.2.1 Giới thiệu về Bootstrap

Bootstrap là một framework mã nguồn mở được phát triển bởi Twitter nhằm hỗ trợ quá trình xây dựng giao diện người dùng (UI) cho các trang web và ứng dụng web. Được công bố lần đầu vào năm 2011, Bootstrap nhanh chóng trở thành một trong những framework phổ biến nhất nhờ tính dễ dùng, khả năng tùy chỉnh cao và hỗ trợ thiết kế responsive (hiển thị tốt trên mọi kích thước màn hình).



Hình 4.2 Bootstrap

4.2.2 Mục đích và vai trò

Mục tiêu chính của Bootstrap là tăng tốc quá trình phát triển giao diện, giúp lập trình viên và nhà thiết kế web:

- Tiết kiệm thời gian khi xây dựng UI.

- Đảm bảo thiết kế nhất quán trên nhiều trình duyệt và thiết bị.
- Dễ dàng tạo các thành phần giao diện đẹp mắt mà không cần viết CSS từ đầu.

4.2.3 Các thành phần chính

Bootstrap cung cấp sẵn rất nhiều thành phần và công cụ hỗ trợ:

a) Hệ thống lưới (Grid System)

- Dựa trên 12 cột, cho phép chia bố cục trang linh hoạt.
- Hỗ trợ responsive theo các kích thước: nhỏ (sm), trung bình (md), lớn (lg), rất lớn (xl), siêu lớn (xxl).

b) Thành phần giao diện (UI Components)

- Các thành phần như: nút (button), biểu mẫu (form), menu (navbar), bảng (table), hộp thoại (modal), cảnh báo (alert),...
- Có thể sử dụng trực tiếp bằng các class CSS có sẵn.

c) Lớp tiện ích (Utility Classes)

- Các class giúp điều chỉnh nhanh: màu sắc, căn lề, khoảng cách, kích thước chữ, độ hiển thị,...
- Giúp tránh việc phải viết CSS riêng lẻ.

d) Tính năng Responsive

- Giao diện tự động thích nghi với các thiết bị khác nhau: điện thoại, máy tính bảng, máy tính để bàn.
- Đảm bảo trải nghiệm người dùng tốt trên mọi nền tảng.

4.3 Thiết kế giao diện hệ thống

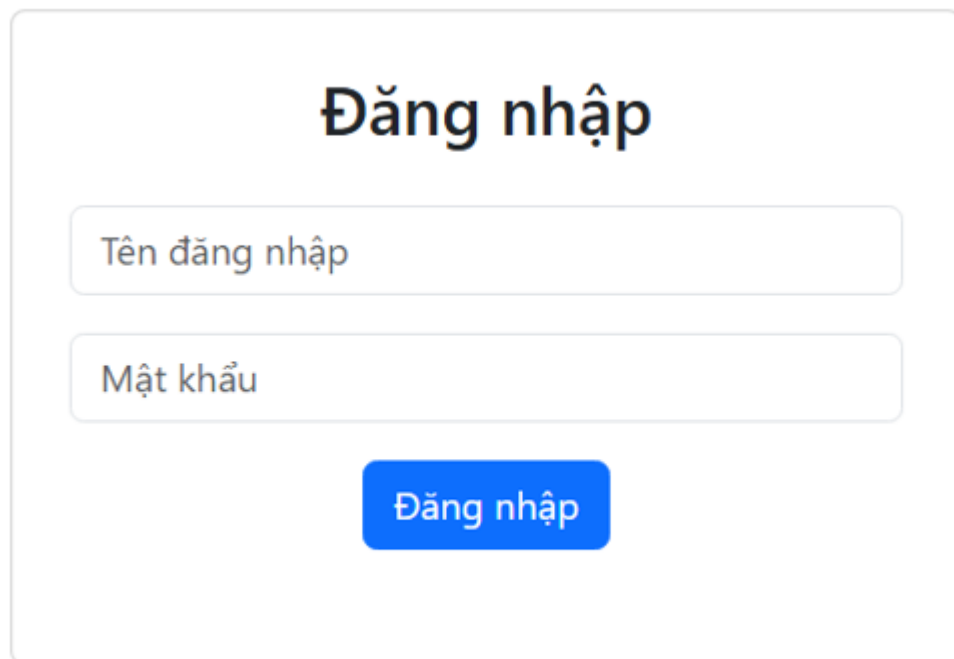
Giao diện hệ thống được thiết kế theo nguyên tắc đơn giản, dễ sử dụng và thân thiện với người dùng, nhằm hỗ trợ hiệu quả quá trình tương tác giữa người dùng và hệ thống. Hệ thống gồm hai giao diện chính: giao diện đăng nhập và giao diện dự đoán hình ảnh u não.

4.3.1 Giao diện đăng nhập

Giao diện đăng nhập là màn hình đầu tiên khi người dùng truy cập vào hệ thống. Tại đây, người dùng cần nhập đầy đủ thông tin tài khoản và mật khẩu để được xác thực quyền truy cập.

Thành phần giao diện:

- Ô nhập Tài khoản.
- Ô nhập Mật khẩu (ẩn ký tự).
- Nút Đăng nhập.
- Thông báo lỗi nếu thông tin chưa hợp lệ.

Hình ảnh minh họa giao diện đăng nhập. Nó bao gồm một khung trắng với tiêu đề "Đăng nhập" ở trung tâm. Dưới tiêu đề là hai ô nhập liệu: "Tên đăng nhập" và "Mật khẩu". Dưới các ô nhập liệu là một nút "Đăng nhập" màu xanh dương.

Hình 4.3 Màn hình đăng nhập

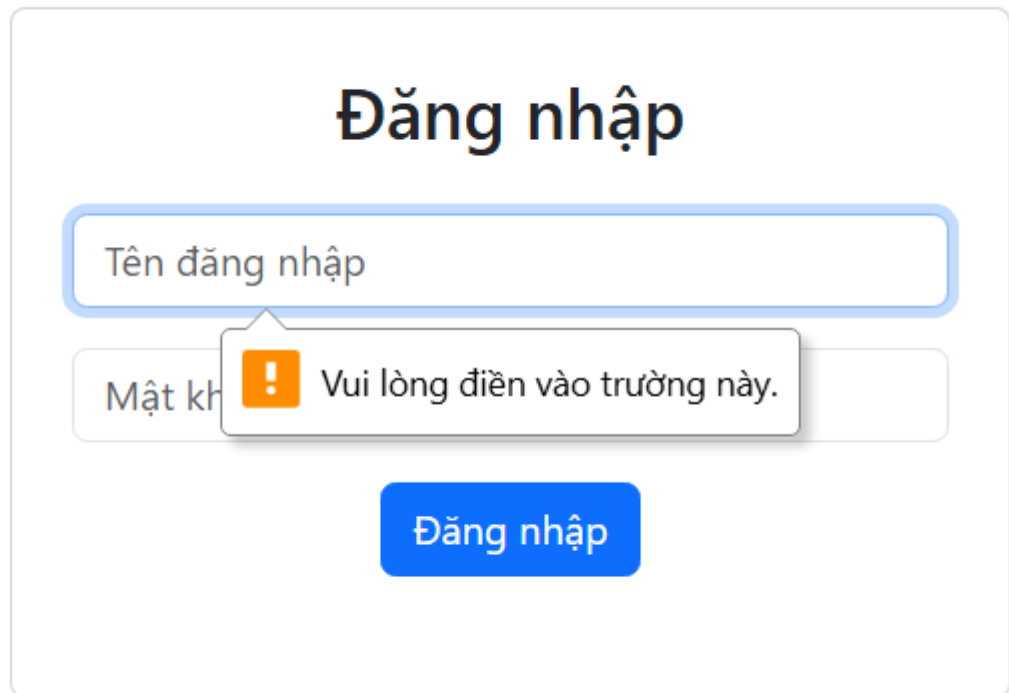
Yêu cầu chức năng:

- Nếu người dùng không nhập tài khoản hoặc mật khẩu → hệ thống hiển thị thông báo:

"Vui lòng nhập đầy đủ tài khoản và mật khẩu."

- Nếu thông tin đúng → hệ thống chuyển đến giao diện dự đoán hình ảnh.
- Nếu thông tin sai → hiển thị thông báo:

"Tài khoản hoặc mật khẩu không đúng"

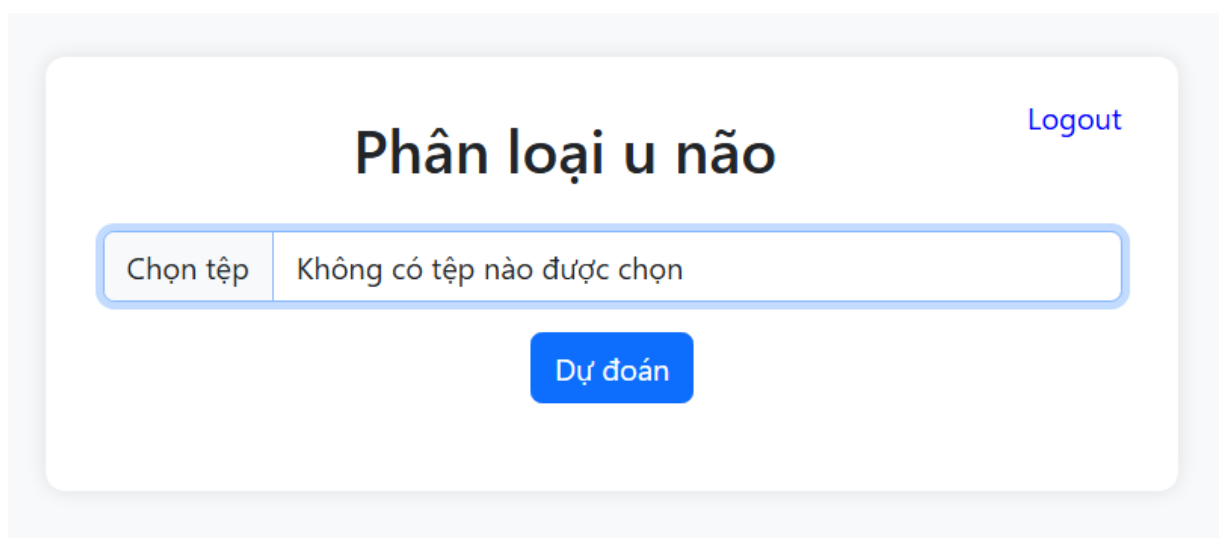


The image shows a login form titled "Đăng nhập". It contains two input fields: "Tên đăng nhập" (Username) and "Mật khẩu" (Password). The "Mật khẩu" field has a validation error message: "Vui lòng điền vào trường này." (Please fill in this field). Below the fields is a blue button labeled "Đăng nhập".

Hình 4.4 Màn hình thông báo yêu cầu đăng nhập

4.3.2 Giao diện dự đoán hình ảnh u não

Sau khi đăng nhập thành công, người dùng sẽ được chuyển đến giao diện dự đoán hình ảnh. Đây là chức năng chính của hệ thống, cho phép người dùng tải lên hình ảnh MRI não và nhận kết quả phân loại loại u.



The image shows a web interface titled "Phân loại u não" (Brain Tumor Classification). In the top right corner, there is a "Logout" link. Below the title is a file upload area with a button labeled "Chọn tệp" (Choose file) and a text box showing "Không có tệp nào được chọn" (No file selected). Below this is a blue button labeled "Dự đoán" (Predict).

Hình 4.5 Màn hình phân loại u não

Thành phần giao diện:

- Nút Chọn ảnh từ thiết bị.
- Khu vực hiển thị hình ảnh đã chọn (nếu có).
- Nút Dự đoán.
- Khu vực hiển thị kết quả (loại u não: glioma, meningioma, pituitary, no tumor).
- Nút Đăng xuất.

[Logout](#)

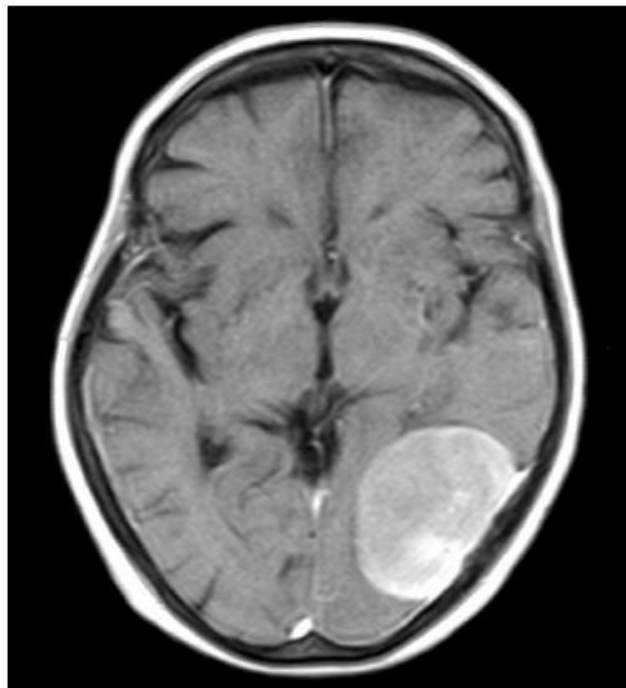
Phân loại u não

Chọn tệp

Không có tệp nào được chọn

Dự đoán

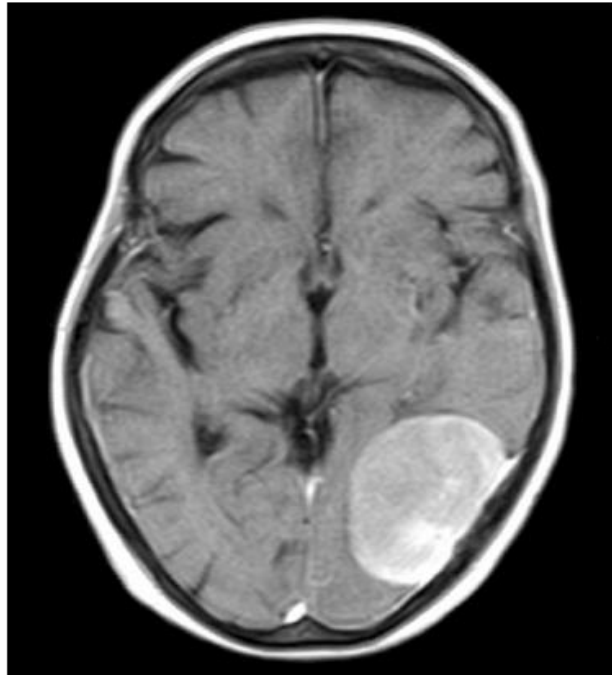
Kết quả:



Dự đoán: **meningioma**

Hình 4.6 Màn hình trả về dự đoán khối u

Kết quả:



Dự đoán: **meningioma**

Thông tin chi tiết

Mô tả: U màng não là khối u phát triển từ màng bao quanh não và tủy sống, phần lớn là lành tính.

Tỉ lệ chữa trị thành công: 90% nếu được phát hiện sớm và điều trị kịp thời.

Phác đồ điều trị: Phẫu thuật loại bỏ u, có thể theo dõi nếu kích thước nhỏ và không có triệu chứng.

Hình 4.7 Màn hình dự đoán và chi tiết khối u

Yêu cầu chức năng:

- Nếu người dùng nhấn nút Dự đoán nhưng chưa chọn ảnh → hiển thị thông báo:

"Vui lòng chọn hình ảnh để dự đoán."

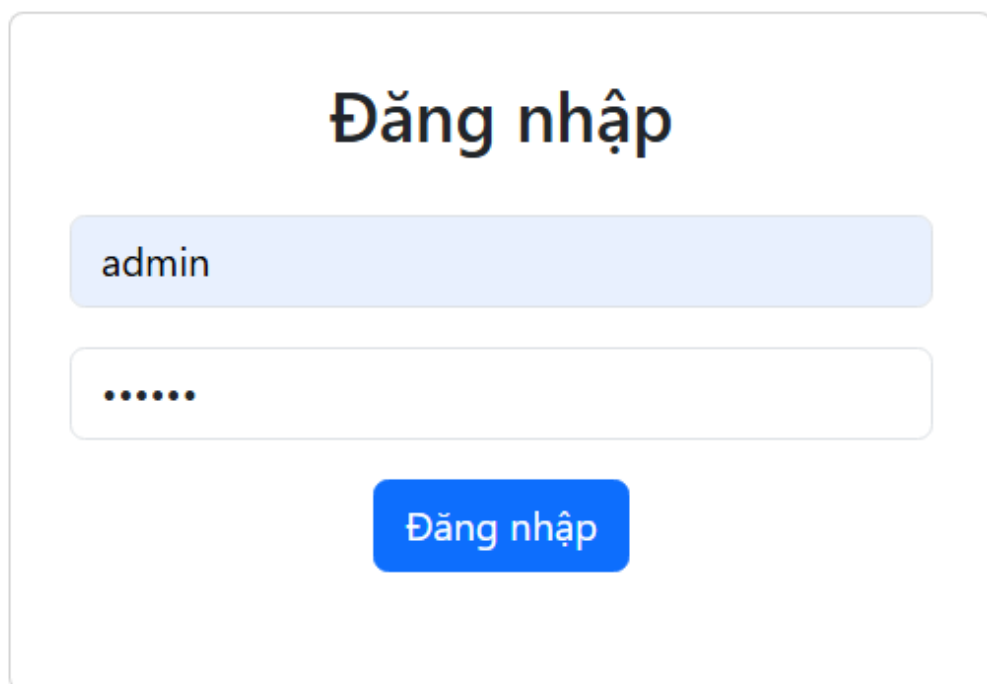
- Nếu đã chọn ảnh hợp lệ → hệ thống thực hiện dự đoán và hiển thị kết quả.
- Nút Đăng xuất cho phép người dùng thoát khỏi hệ thống và quay lại giao diện đăng nhập.

4.4 Kiểm thử

4.4.1 Kiểm thử giao diện đăng nhập

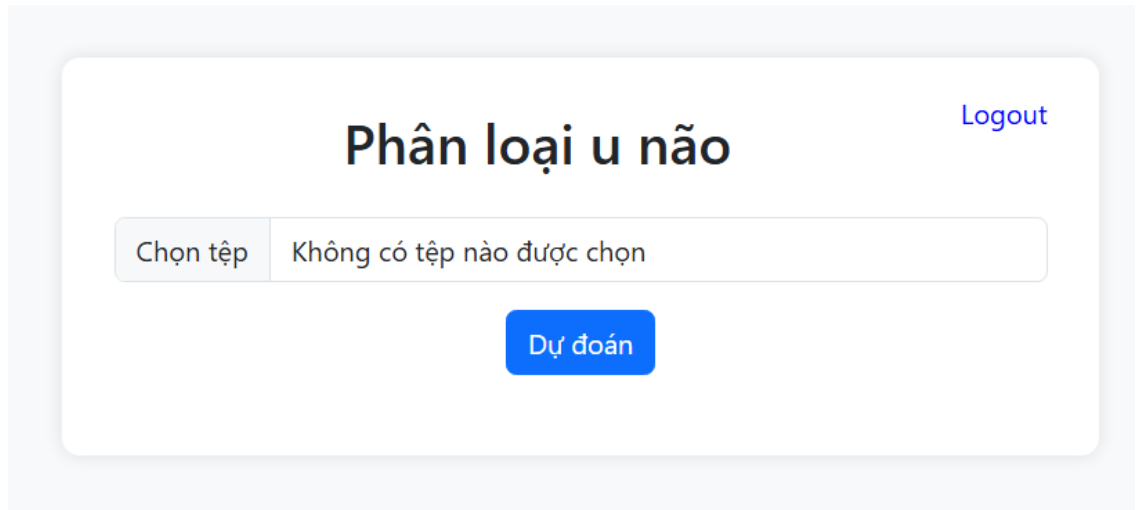
a) Kịch bản 1: Đăng nhập thành công

Mục tiêu của kiểm thử này là đảm bảo rằng hệ thống cho phép người dùng đăng nhập thành công khi cung cấp đúng thông tin tài khoản và mật khẩu. Người kiểm thử truy cập vào trang đăng nhập, nhập tên đăng nhập và mật khẩu hợp lệ (ví dụ: "admin" và "123456"), sau đó nhấn nút "Đăng nhập". Hệ thống cần chuyển hướng người dùng đến giao diện dự đoán hình ảnh mà không xuất hiện bất kỳ thông báo lỗi nào.



The image shows a login interface with a light blue header containing the text "Đăng nhập". Below the header, there are two input fields. The first input field contains the text "admin". The second input field contains six dots, indicating a password. Below the input fields, there is a blue button with the text "Đăng nhập".

Hình 4.8 Màn hình đăng nhập



Phân loại u não

Logout

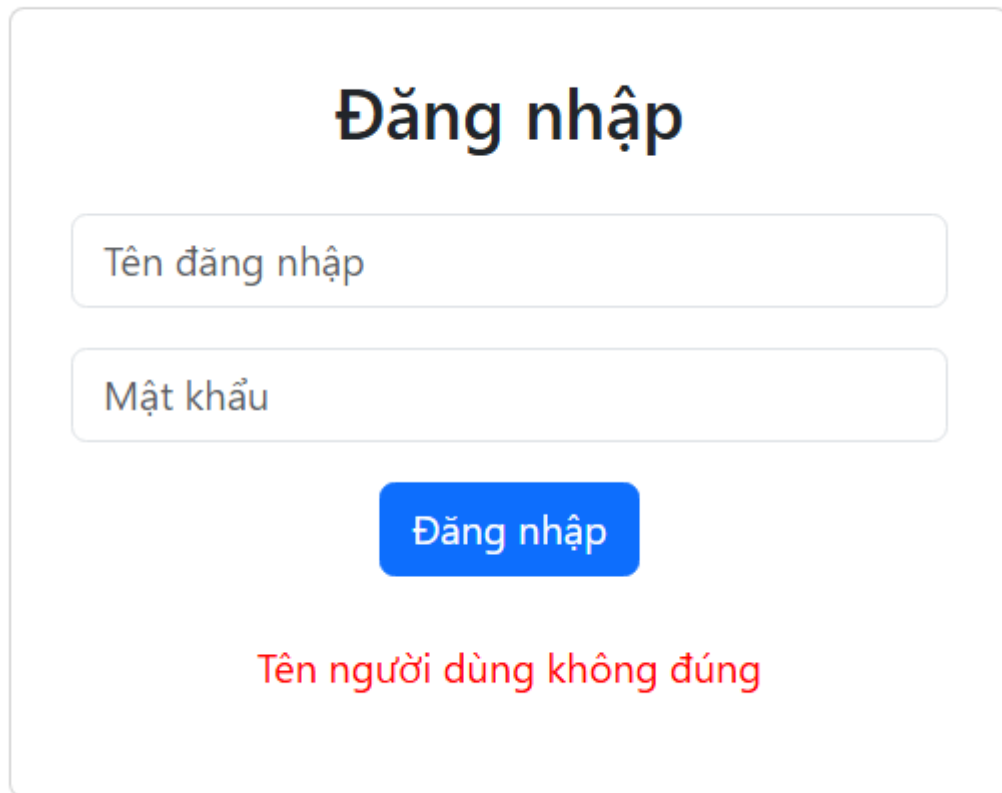
Chọn tệp Không có tệp nào được chọn

Dự đoán

Hình 4.9 Màn hình sau khi đăng nhập thành công

b) Kịch bản 2: Đăng nhập thất bại khi sai thông tin

Kiểm thử nhằm xác minh hệ thống xử lý chính xác khi người dùng nhập sai mật khẩu hoặc tên đăng nhập. Người kiểm thử nhập một trong hai thông tin không chính xác và nhấn “Đăng nhập”. Hệ thống phải hiện thông báo lỗi như: “Sai tên đăng nhập hoặc mật khẩu”, không cho phép truy cập vào hệ thống.



The image shows a login interface with a title, two input fields, a submit button, and an error message.

Đăng nhập

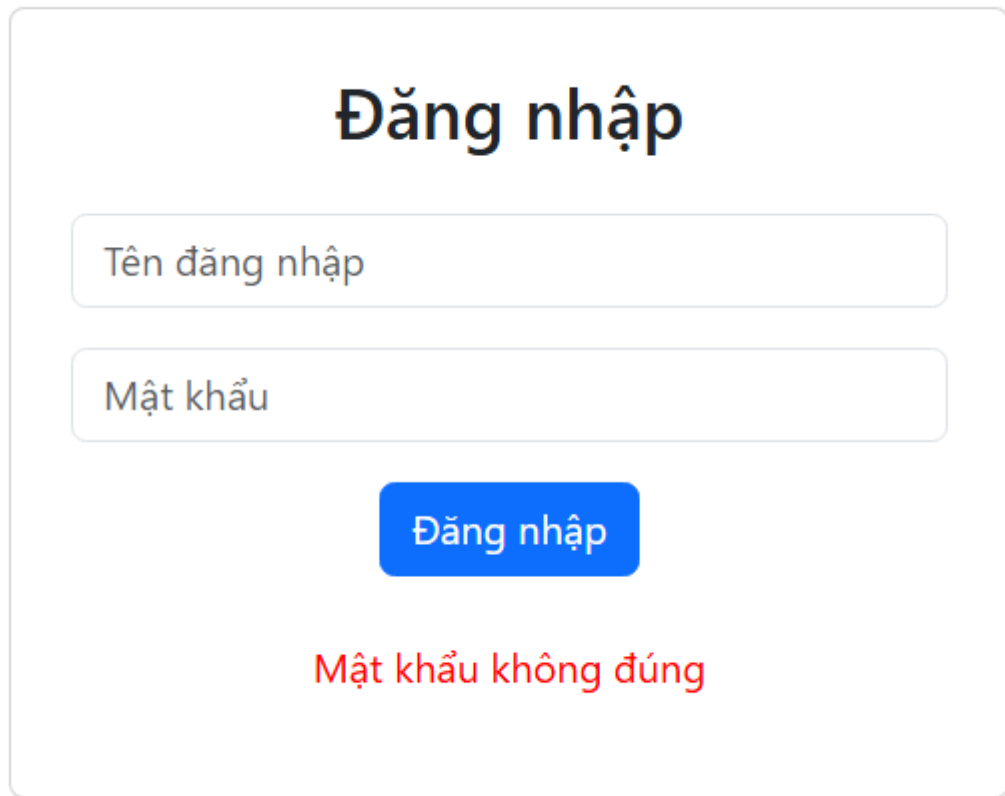
Tên đăng nhập

Mật khẩu

Đăng nhập

Tên người dùng không đúng

Hình 4.10 Màn hình khi đăng nhập sai tên người dùng



Đăng nhập

Tên đăng nhập

Mật khẩu

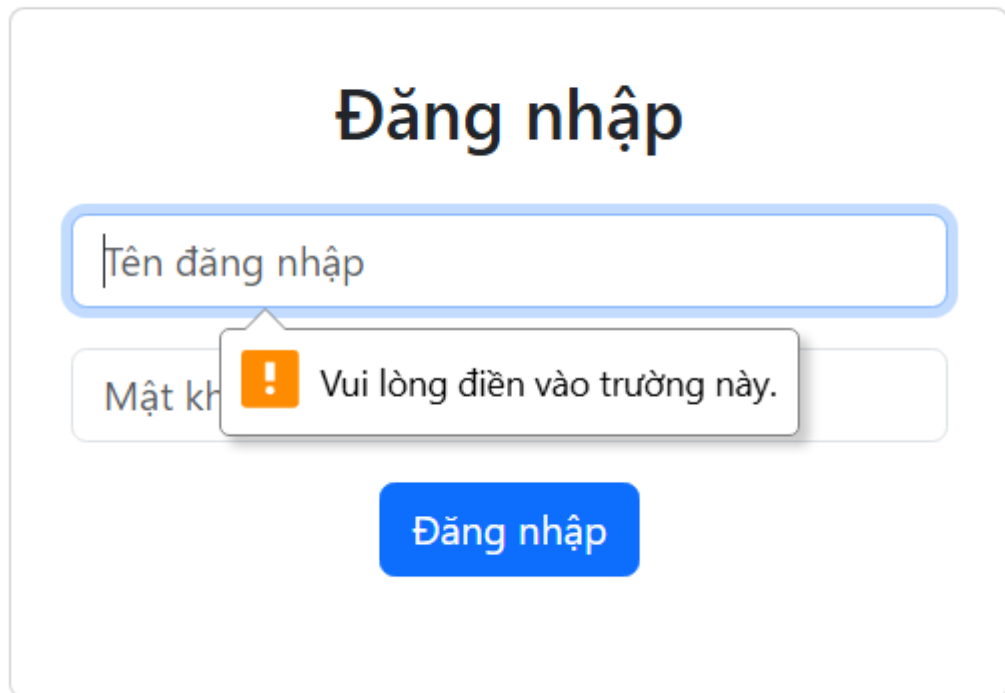
Đăng nhập

Mật khẩu không đúng

Hình 4.11 Màn hình khi nhập sai mật khẩu

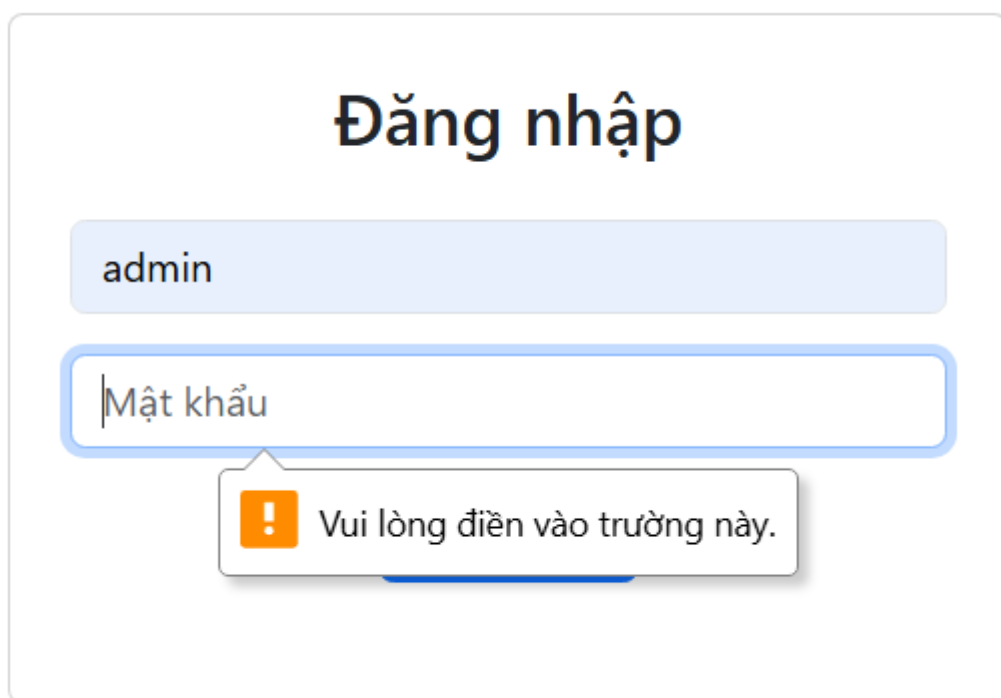
c) Kịch bản 3: Đăng nhập với trường thông tin trống

Mục tiêu là đảm bảo hệ thống không cho phép người dùng đăng nhập khi chưa điền đủ thông tin. Khi người kiểm thử để trống cả hai trường hoặc chỉ điền một trường, nhấn “Đăng nhập”, hệ thống cần hiển thị thông báo yêu cầu điền đầy đủ thông tin, ví dụ: “Vui lòng nhập đầy đủ tên đăng nhập và mật khẩu”.



The image shows a login form titled "Đăng nhập". It has two input fields: "Tên đăng nhập" (Username) and "Mật khẩu" (Password). The "Tên đăng nhập" field is empty and has a blue border. The "Mật khẩu" field is also empty but has a grey border. A tooltip with an orange exclamation mark icon and the text "Vui lòng điền vào trường này." (Please fill in this field.) points to the "Mật khẩu" field. Below the fields is a blue button labeled "Đăng nhập".

Hình 4.12 Màn hình đăng nhập với thông tin trống tên đăng nhập



The image shows a login form titled "Đăng nhập". It has two input fields: "admin" (Username) and "Mật khẩu" (Password). The "admin" field is filled with the text "admin" and has a light blue background. The "Mật khẩu" field is empty and has a blue border. A tooltip with an orange exclamation mark icon and the text "Vui lòng điền vào trường này." (Please fill in this field.) points to the "Mật khẩu" field. Below the fields is a blue button labeled "Đăng nhập".

Hình 4.13 Màn hình đăng nhập với thông tin trống mật khẩu

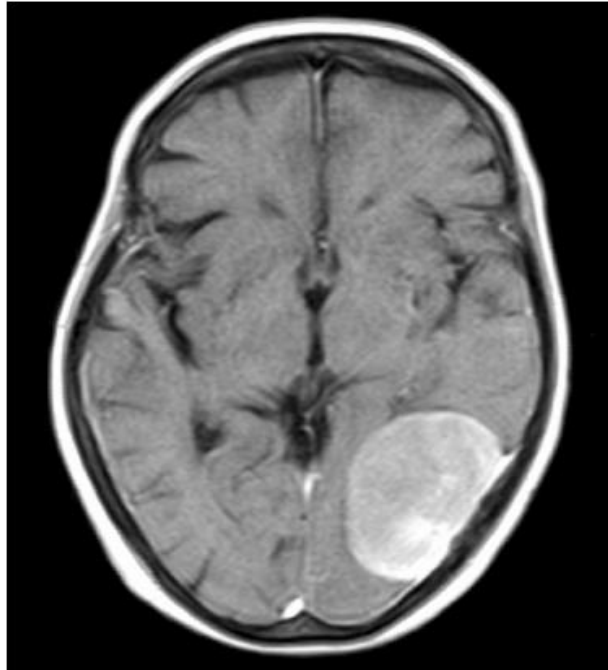
4.4.2 Kiểm thử giao diện dự đoán hình ảnh u não

a) Kịch bản 1: Dự đoán ảnh thành công

Người dùng sau khi đăng nhập thành công sẽ truy cập giao diện dự đoán. Tại đây, người dùng chọn một ảnh MRI não hợp lệ từ thiết bị và nhấn nút “Dự đoán”. Hệ thống cần xử lý ảnh, chạy mô hình học sâu và trả về kết quả dự đoán (ví dụ: meningioma, glioma, pituitary hoặc không có u). Đồng thời, ảnh và nhãn dự đoán cần được hiển thị trên màn hình.

Dự đoán

Kết quả:



Dự đoán: **meningioma**

Thông tin chi tiết

Mô tả: U màng não là khối u phát triển từ màng bao quanh não và tủy sống, phần lớn là lành tính.

Tỉ lệ chữa trị thành công: 90% nếu được phát hiện sớm và điều trị kịp thời.

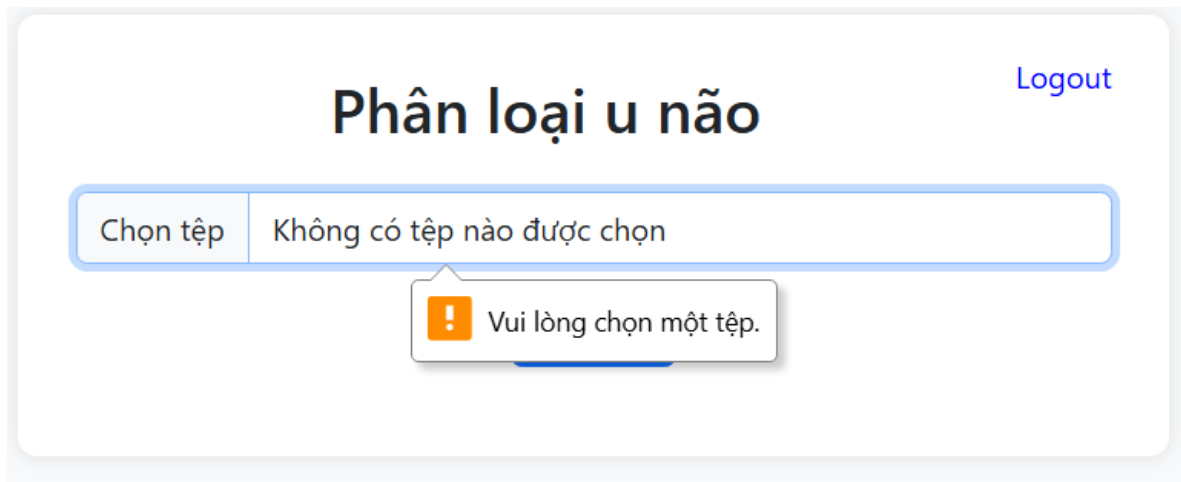
Phác đồ điều trị: Phẫu thuật loại bỏ u, có thể theo dõi nếu kích thước nhỏ và không có triệu chứng.

Hình 4.14 Màn hình dự đoán thành công

b) Kịch bản 2: Không chọn ảnh nhưng nhấn dự đoán

Mục tiêu là kiểm tra phản hồi của hệ thống khi người dùng chưa chọn ảnh nhưng cố gắng dự đoán. Khi nhấn nút “Dự đoán” mà không tải ảnh, hệ thống

phải cảnh báo hoặc hiện thông báo lỗi, ví dụ: “Vui lòng chọn ảnh trước khi thực hiện dự đoán”.



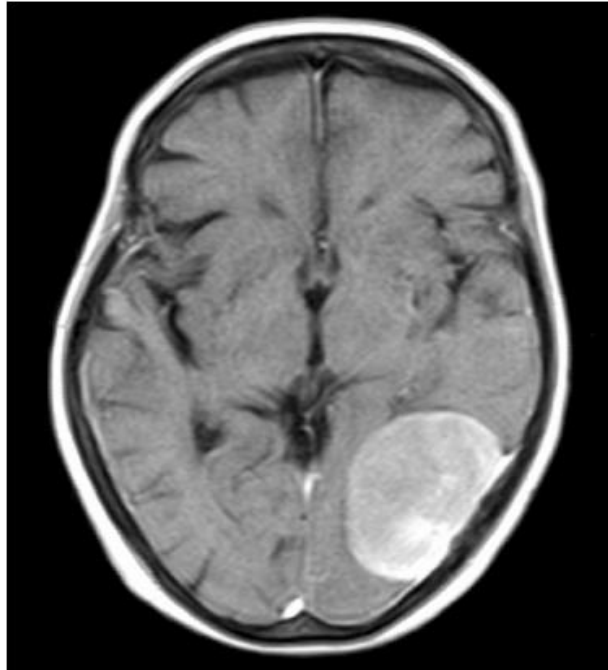
Hình 4.15 Màn hình dự đoán nhưng không chọn ảnh .

c) Kịch bản 3: Hiển thị kết quả dự đoán

Sau khi dự đoán thành công, giao diện cần thể hiện rõ ràng ảnh mà người dùng đã chọn cùng với nhãn kết quả. Kịch bản kiểm thử này đảm bảo rằng giao diện dễ quan sát, kết quả rõ ràng, tránh gây nhầm lẫn cho người sử dụng.

Dự đoán

Kết quả:



Dự đoán: **meningioma**

Thông tin chi tiết

Mô tả: U màng não là khối u phát triển từ màng bao quanh não và tủy sống, phần lớn là lành tính.

Tỉ lệ chữa trị thành công: 90% nếu được phát hiện sớm và điều trị kịp thời.

Phác đồ điều trị: Phẫu thuật loại bỏ u, có thể theo dõi nếu kích thước nhỏ và không có triệu chứng.

Hình 4.16 Màn hình hiển thị kết quả dự đoán

4.5 Tóm tắt chương

Chương 4 trình bày quá trình xây dựng sản phẩm ứng dụng web hỗ trợ nhận dạng ảnh u não dựa trên mô hình học sâu. Trước tiên, chương giới thiệu Flask – một micro framework Python được sử dụng để xây dựng backend của hệ thống,

nhờ tính linh hoạt, dễ triển khai và khả năng tích hợp tốt với các mô hình học máy. Bên cạnh đó, Bootstrap được sử dụng cho phần frontend nhằm tạo ra giao diện thân thiện, hiện đại và dễ sử dụng.

Tiếp theo, chương trình bày chi tiết về thiết kế giao diện hệ thống, bao gồm trang đăng nhập và giao diện dự đoán ảnh u não – nơi người dùng có thể tải ảnh lên và nhận kết quả dự đoán ngay trên nền web. Việc kết hợp giữa mô hình học sâu và giao diện trực quan đã tạo ra một sản phẩm có tính ứng dụng cao, góp phần hỗ trợ bác sĩ hoặc người dùng trong công tác chẩn đoán hình ảnh y tế.

KẾT LUẬN

Thời gian làm đồ án vừa qua là trải nghiệm vô cùng thú vị và đáng giá với tôi. Tôi đã tìm hiểu các kỹ thuật khác nhau của trí tuệ nhân tạo nhằm giải quyết bài toán phân loại u não, qua đó không chỉ tích lũy được các kinh nghiệm về chuyên môn mà còn học được cái kỹ năng làm việc độc lập, quản lý thời gian. Đây chắc chắn sẽ là hành trang quý giá trong sự nghiệp tương lai của tôi.

Tôi đã tìm hiểu, nghiên cứu, ứng dụng các mô hình học máy cũng như tận dụng các nghiên cứu từng được công bố về u não để hoàn thành đề tài Nghiên cứu các kỹ thuật của Trí tuệ nhân tạo cùng các mô hình học máy Deep Learning, và ứng dụng cho bài toán hỗ trợ chuẩn đoán u não. Hệ thống đã cho thấy những thành công bước đầu khi thu được những kết quả nhận dạng với độ chính xác tốt trên. Hy vọng rằng nghiên cứu của tôi có thể góp phần thúc đẩy các đề tài liên quan tới hỗ trợ và phát triển y tế nước nhà.

Hiện tại, hệ thống vẫn còn một số hạn chế như độ chính xác nhận dạng chưa cao ở một số trường hợp phức tạp, chưa tích hợp được các chức năng nâng cao như phân đoạn khối u hay xác định mức độ nghiêm trọng. Ngoài ra, việc triển khai ứng dụng thành một sản phẩm thực tế sử dụng trong môi trường lâm sàng vẫn chưa được thực hiện.

Trong thời gian tới, tôi sẽ tiếp tục cải tiến mô hình bằng cách thử nghiệm các phương pháp nâng cao như attention mechanism hoặc các mô hình transformer. Tôi cũng dự định mở rộng tập dữ liệu, cải thiện giao diện người dùng và hướng tới triển khai ứng dụng trên nền tảng đám mây để tiện lợi hơn cho việc sử dụng và thử nghiệm thực tế.

TÀI LIỆU THAM KHẢO

- [1] Báo lao động thủ đô thống kê tình hình tham gia bảo hiểm y tế. URL: <https://laodongthudo.vn/ca-nuoc-co-9552-trieu-nguoi-tham-gia-bao-hiem-y-te-dat-942-dan-so-182831.html>. Lần truy cập gần nhất ngày 17/05/2025
- [2] Bệnh viện hữu nghị Việt Đức báo cáo tình hình ung thư não <https://benhvienvietduc.org/hon-2-500-nguoi-benh-u-nao-duoc-kham-va-dieu-tri-moi-nam-tai-benh-vien-huu-nghi-viet-duc.html>. Lần truy cập gần nhất ngày 17/05/2025
- [3] Globocan 2022 tình hình ung thư não. <https://lifestyle.znews.vn/u-nao-can-benh-cua-niem-dau-va-cuoc-chien-sinh-tu-post1536178.html>. Lần truy cập gần nhất ngày 17/05/2025
- [4] Bệnh viện Ung bướu TP.HCM thống kê tình hình ung thư não <https://thanhnien.vn/benh-u-nao-co-dieu-tri-duoc-khong-nhung-bien-phap-chu-dong-phong-ngua-185250319145242373.htm>. Lần truy cập gần nhất ngày 17/05/2025
- [5] Mô hình K-Means. URL: <https://machinelearningcoban.com/2017/01/01/kmeans/>. Lần truy cập gần nhất ngày 17/05/2025
- [6] Mạng CNN. URL: <https://vietnix.vn/neural-network/>. Lần truy cập gần nhất ngày 17/05/2025
- [7] Hàm kích hoạt. URL: <https://medium.com/@meetkp/understanding-the-rectified-linear-unit-relu-a-key-activation-function-in-neural-networks-28108fba8f07>. Lần truy cập gần nhất ngày 17/05/2025
- [8] Hàm kích hoạt. URL: <https://zhuanlan.zhihu.com/p/584399939>. Lần truy cập gần nhất ngày 17/05/2025

[9] Hàm kích hoạt. URL: <https://bizflycloud.vn/tin-tuc/hoi-quy-tuyen-tinh-va-hoi-quy-logistic-khac-nhau-o-diem-nao-20240617142629797.htm>. Lần truy cập gần nhất ngày 17/05/2025

[10] Hàm kích hoạt. URL: <https://medium.com/@minhducnguyen20/neural-networks-activation-functions-3892f6cc1694>. Lần truy cập gần nhất ngày 17/05/2025

[11] Tầng kết nối đầy đủ. URL: <https://medium.com/data-science/convolutional-layers-vs-fully-connected-layers-364f05ab460b>. Lần truy cập gần nhất ngày 17/05/2025

[12] Mạng ResNet50. URL: <https://www.sciencedirect.com/science/article/pii/S2667143324000738>. Lần truy cập gần nhất ngày 17/05/2025

[13] Mạng DenseNet. URL: <https://thanahongsuwan.medium.com/%E0%B8%A1%E0%B8%B2%E0%B8%AA%E0%B8%A3%E0%B9%89%E0%B8%B2%E0%B8%87-densenet-%E0%B9%82%E0%B8%94%E0%B8%A2%E0%B9%83%E0%B8%8A%E0%B9%89-pytorch-%E0%B8%81%E0%B8%B1%E0%B8%99-893c94b7a1e6>.
Lần truy cập gần nhất ngày 17/05/2025