ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

Báo cáo

ĐỒ ÁN TRÍ TUỆ NHÂN TẠO

Đề tài:

Phát hiện và phân loại ung thư phổi thông qua ảnh chụp cắt lớp CT

GVHD: Trần Huy

SV thực hiện:

Lê Nguyễn Hải Đăng	2113176
Đinh Vũ Hà	2113269
Từ Mai Thế Nhân	2114277
Trần Bảo Phúc	2114452

TP. HỒ CHÍ MINH, THÁNG 10/2023

Mục lục

I Lời cảm ơn

Lời đầu tiên, nhóm tác giả xin gửi lời cảm ơn đến các quý thầy cô khoa Khoa học và Kỹ thuật Máy Tính, trường đại học Bách Khoa TPHCM đã tạo cơ hội cho được học tập, rèn luyện và tích lũy kiến thức, kỹ năng để thực hiện bài nghiên cứu. Đặc biệt, nhóm xin gửi lời cảm ơn đến giảng viên hướng dẫn thầy Trần Huy đã tận tình chỉ dẫn, theo dõi và đưa ra những lời khuyên bổ ích giúp nhóm hướng tới một kết quả nghiên cứu toàn diện, chính xác hơn. Cuối cùng, nhóm tác giả xin chân thành cảm ơn gia đình, bạn bè đã luôn động viên và tạo điều kiện tốt nhất để nhóm có thể nỗ lực hoàn thành tốt bài nghiên cứu.

II Tổng quan về nội dung

Phát hiện và chẩn đoán ung thư sớm là việc vô cùng quan trọng trong y học để tăng cơ hội điều trị và cải thiện chất lượng cuộc sống của bệnh nhân. Trong bài báo này, nhóm tác giả tập trung vào sử dụng bài toán phân vùng ảnh để giúp phát hiện tế bào ung thư não một cách nhanh chóng và hiệu quả hơn từ tập ảnh **CT** được cung cấp. Phương pháp nghiên cứu trong bài báo tập trung vào xây dựng một mô hình mạng học sâu để phân tích ảnh. Cụ thể, nhóm tác giả đã xây dựng một mô hình sử dụng mạng **ResNet** để phân vùng tế bào ung thư phổi từ tập ảnh 2D được cắt ra từ khối **CT** cho trước. Các kết quả thu được cho thấy mô hình này cho dự đoán khá tốt trong việc phát hiện các khối u. Tuy nhiên, với những trường hợp khối u phức tạp, mô hình cho kết quả với độ chính xác không cao. Kết luận, mặc dù đây chưa phải là phương pháp cho tỉ lệ chính xác cao nhất trong thời điểm hiện tại nhưng mô hình sử dụng mạng **ResNet** trong bài báo này đã cung cấp một giải pháp hiệu quả cho việc phát hiện và chẩn đoán tế bào ung thư phổi, góp phần cải thiện cuộc sống của con người.

III Giới thiệu chung

1 Danh sách thành viên & phân chia công việc

2 Giới thiệu đề tài

2.1 Tính cấp thiết của đề tài

Ở thời điểm hiện tại, ung thư là một trong những nguyên nhân phổ biến hàng đầu gây tử vong đối với nhân loại. Theo ước tính, có khoảng 20 triệu người ca ung thư được phát hiện trên toàn thế giới vào năm 2023, trong đó khoảng 10 triệu ca tử vong và con số này được dự đoán rằng sẽ không ngừng tăng lên. Ung thư là sự phát triển bất thường của một nhóm các tế bào, có khả năng xâm nhập và hủy hoại các tế bào hay mô khác và có thể xảy ra ở bất kì cơ quan nào trên cơ thể. Song, nếu được phát hiện từ sớm, hầu hết các loại ung thư đều có thể chữa trị được. Ở các trung tâm nghiên cứu và các bệnh viện ngày nay, việc chẩn đoán các tế bào ung thư hầu như được thực hiện thủ công bởi các chuyên gia. Điều này gây nên lãng phí về nhân lực, và chi phí rất tốn kém.

Trong số đó, ung thư phổi là một trong những loại ung thư có tỉ lệ tử vong cao nhất, là một loại bệnh hiểm nghèo và nguy hiểm trong hệ thống hô hấp của con người. Nó bắt đầu khi các tế bào trong phổi bị biến đổi gen và phát triển không kiểm soát. Ung thư phổi thường không gây ra triệu chứng trong giai đoạn đầu, điều này khiến việc phát hiện bệnh trở nên khó khăn. Nguyên nhân thường bắt nguồn từ những thói quen sinh hoạt không tốt, tiếp xúc nhiều với khói thuốc, khí độc và ô nhiễm môi trường. Theo thống kê 9 tháng đầu năm 2023, đã có 238,340 ca mắc ung thư phổi ở Mỹ và 2,206,771 ca trên toàn thế giới. Việc chẩn đoán ung thư phổi thường đòi hỏi quá trình xét nghiệm như chụp X-quang, siêu âm và CT-scan phổi để xác định kích thước và vị trí của khối u.

Chính vì vậy, một phương pháp đưa ra dự báo chính xác về các khối u gây ra ung thư một cách tự động sẽ giúp giảm được rất nhiều chi phí cho việc chẩn đoán ung thư, vốn là một trong những vấn đề cũng được nhiều chuyên gia quan tâm ngày nay.

2.2 Thông tin chung về đối tượng nghiên cứu

2.2.1 Đối tượng nghiên cứu

Nhận thấy vấn đề cấp thiết và mang lại ý nghĩa cao như đã phân tích trước đó, nhóm tác giả đã quyết định lựa chọn bài toán phân loại ảnh để phát hiện các giai đoạn của tế bào ung thư (cụ thể ở đây là các tế bào ung thư phổi) từ một tập ảnh **CT** cho trước. Thông qua dự án, nhóm tác giả hi vọng có thể xây dựng được một mô hình có thể phát hiện được các tế bào ung thư, mang lại ý nghĩa về mặt y tế.

2.2.2 Phạm vi nghiên cứu

Nhóm tác giả nghiên cứu đề tài về phát hiện các tế bào ung thư phổi trong phạm vi cụ thể là tại các trung tâm, viện nghiên cứu và các bệnh viện, cơ sở y tế, cụ thể là phòng ban chuyên về xử lý ảnh y tế và các khoa Chẩn đoán hình ảnh, và trong khoảng thời gian là 4 tháng.

2.3 Mục tiêu nghiên cứu

2.3.1 Mục tiêu nghiên cứu tổng quát

Thông qua bài nghiên cứu này, nhóm tác giả mong muốn có thể nâng cao tỷ lệ chính xác trong việc chẩn đoán chính xác của bệnh ung thư và giúp việc chẩn đoán trở nên dễ dàng hơn,

nhanh hơn cho các chuyên gia y tế, cũng như đáng tin cậy hơn đối với bệnh nhân, giảm bớt gánh nặng cho các chuyên gia khi phải xác định một khối u (đặc biệt là khối u ở phổi), từ đó giúp cải thiện đời sống con người và nâng cao khả năng đẩy lùi bệnh ung thư.

2.3.2 Mục tiêu nghiên cứu cụ thể

- Thứ nhất, tiếp xúc với bài toán phân vùng ảnh nói chung và bài toán phân vùng ảnh trong lĩnh vực y khoa nói riêng, đồng thời, nắm bắt được các khái niệm hỗ trợ giải quyết bài toán như lớp tích chập, lớp gộp và các hàm kích hoạt phi tuyến...
- Thứ hai, nắm được các bước xây dựng một mô hình học sâu, từ giai đoạn tiền xử lí dữ liệu đến giai đoạn huấn luyện cũng như đánh giá được mô hình thông qua các thang đo độ chính xác.
- Thứ ba, xây dựng được một mô hình phân loại được 3 trạng thái của tế báo ung thư (bình thường, mới bắt đầu, ác tính).
- Thứ tư, mô hình được xây dựng mang lại hiệu quả có thể so sánh được về chi phí so với các mô hình hiện có.
- Thứ năm, đánh giá tổng quan về kết quả đã đạt được trong dự án của nhóm. Từ đó rút ra những kinh nghiệm cần thiết cho nhóm tác giả trong quá trình phát triển nâng cao mô hình hiện có hoặc phát triển các mô hình khác.

2.4 Các kết quả cần đạt được

Nhóm tác giả mong muốn khi hoàn thành bài nghiên cứu có thể đạt được một số kết quả sau đây, cụ thể:

- Có cái nhìn tổng quan về bài toán phân loại ảnh, nắm được các khái niệm liên quan và hỗ trợ cho việc tiếp cận bài toán.
- Xây dựng được một mô hình sử dụng mô hình mạng **ResNet** để phân biệt được 3 trạng thái cơ bản của tế bào ung thư phổi từ hình ảnh chụp **CT** từ trước.
- Đánh giá được độ tốt mô hình thông qua các chỉ số trong thang đo độ chính xác.
- Đánh giá được những mặt làm được và những hạn chế cần khắc phục, đồng thời đề ra được những hướng đi tiếp theo.

IV Cơ sở lý thuyết

1 Bài toán phân loại ảnh

1.1 Về bài toán tổng quát

Phân loại hình ảnh (Image classification) hay Nhận dạng hình ảnh (Image recognition) là một trong những tác vụ của thị giác máy tính, ở đó thuật toán xem xét và dán nhãn cho hình ảnh từ một tập danh mục được xác định và đào tạo trước.

Ví dụ, với một tập các hình ảnh, mỗi hình ảnh mô tả một con mèo hoặc một con chó, thuật toán sẽ "quan sát" toàn bộ dữ liệu và dựa trên hình dạng, màu sắc để hình thành giả thuyết liên quan đến nội dung của ảnh. Kết quả thu được là từ tập dữ liệu ban đầu, các hình ảnh chó/mèo đã được phân loại một cách tự động.

Thực tế, thị giác góp phần tạo nên 80-85 % nhận thức của con người về thế giới. Hàng ngày, mỗi người phải thực hiện phân loại trên bất kỳ dữ liệu hình ảnh nào mà chúng ta bắt gặp.

Do đó, mô phỏng nhiệm vụ phân loại với sự trợ giúp của mạng nơ-ron là một trong những ứng dụng đầu tiên của thị giác máy tính mà các nhà nghiên cứu nghĩ đến.

1.2 Bài toán trong lĩnh vực y khoa

Việc phát hiện ung thư tự động đã được thực hiện và sẽ trở nên phổ biến hơn. Chẩn đoán hỗ trợ bằng máy tính (CAD) đang ngày càng phát triển, và việc phát hiện và phân loại ung thư đã đạt được trong việc xác định các dạng phụ của bệnh bạch cầu với mạng lưới thần kinh chập dày đặc và mạng nơ-ron phức hợp. Một hệ thống chẩn đoán hỗ trợ bằng máy tính với mạng lưới thần kinh nhân tạo khổng lồ dựa trên kỹ thuật mô mềm đã phát hiện ung thư phổi trong hình ảnh X-quang. Sự lây nhiễm của Helicobacter pylori đã được dự đoán bằng hình ảnh nội soi bằng trí tuệ nhân tạo. Một mạng lưới thần kinh chập dựa trên vùng nhanh hơn đã được áp dụng để chẩn đoán giai đoạn T của ung thư dạ dày trong hình ảnh chụp cắt lớp vi tính nâng cao (CT) của ung thư dạ dày. Hình ảnh kỹ thuật số của dữ liệu bệnh lý trong ung thư đã được sử dụng trong chẩn đoán ung thư. Phân tích bệnh lý bằng kỹ thuật số sử dụng hình ảnh toàn bộ các lát cắt có thể góp phần vào đánh giá "từ xa". Phân tích hình ảnh tự động và các ứng dụng trí tuệ nhân tạo ngày càng tăng trong lĩnh vực bệnh lý tuyến giáp. Việc nhận dạng ung thư bằng trí tuệ nhân tạo đã trở nên chính xác và chính xác hơn, đi kèm với sự tiến bộ của mạng lưới thần kinh và khả năng tính toán.

Quy trình làm việc lâm sàng nâng cao với các can thiệp trí tuệ nhân tạo đã được đề xuất trong điều trị ung thư, bao gồm phát hiện và xác định đặc điểm do trí tuệ nhân tạo hướng dẫn, lập kế hoạch và theo dõi điều trị do trí tuệ nhân tạo hướng dẫn và tối ưu hóa kết quả theo định hướng trí tuệ nhân tạo. Các công cụ trí tuệ nhân tạo có thể được sử dụng để phát hiện các bất thường, xác định đặc điểm của tổn thương nghi ngờ và xác định tiên lượng hoặc đáp ứng với điều trị. Công nghệ trí tuệ nhân tạo cung cấp các bộ mô tả khối u mạnh mẽ trong phân đoạn, chẩn đoán, phân giai đoạn và hình ảnh gen. Trích xuất đặc điểm phóng xạ từ hình ảnh CT Scan của bệnh nhân ung thư phổi đã thành công để cho thấy mối liên quan với biểu hiện gen và khả năng tiên lượng. Các đặc điểm chụp X quang dựa trên CT có thể dự đoán di căn xa cho bệnh nhân ung thư biểu mô tuyến phổi.

2 Các nghiên cứu liên quan

2.1 Lớp tích chập (Convolutional Layer)

Lớp tích chập, hay convolutional layer, vốn được sử dụng chủ yếu trong các mạng neuron tích chập (Convolutional Neural Networks) để giúp mô hình có thể xử lý được những dạng dữ liệu có cấu trúc đã được biết trước và có dạng lưới (grid-like topology). Một số ví dụ điển hình cho những dữ liệu có dạng này bao gồm dữ liệu về chuỗi thời gian (time series data - lưới 1D), hay ảnh số (digital image - lưới 2D), v.v. Và nguồn gốc của ý tưởng về lớp tích chập bắt nguồn từ một phép tính trong toán học, đó là phép tích chập.

Ở dạng tổng quát, tích chập là một phép tính trên hai hàm thực, được định nghĩa như sau: Với x(t) là phép đo một đại lượng theo thời gian và gọi w(τ) là một hàm trọng số (thường là một hàm phân phối xác suất) và τ là khoảng thời gian đã trôi qua của phép đo, khi đó:

$$s(t) = \int x(\tau)w(t-\tau)dx \tag{1}$$

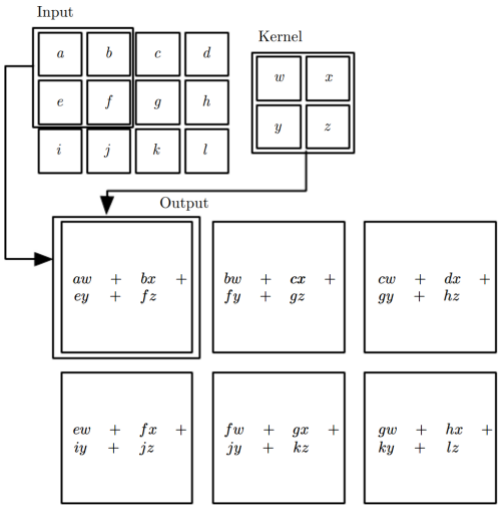
Về mặt ý nghĩa, hàm s(t) trong phương trình (1) là một ước lượng trơn của hàm x(t) [6]. Như vậy, nếu xét theo góc nhìn từ học máy, thì khi sử dụng phép tính này, ta sẽ nhận được một

đầu ra ít nhiễu, và mượt hơn so với đầu vào lúc ban đầu.

Và, nếu ta định nghĩa rằng x và w chỉ được xác định với các giá trị nguyên t, ta có thể định nghĩa phép tích chập rời rạc:

$$s(t) = (x * \tau)(t) = \sum_{-\infty}^{\infty} x(\tau)w(t-\tau) \tag{2}$$

Với một số chỉnh sửa để đáp ứng cho việc tính toán trên các tensor, phương trình (2) sẽ là khởi đầu cho ý tưởng ứng dụng phép tính này vào mạng học sâu. Để có thể hình dung, nhóm tác giả sẽ minh họa phép tính này với một ma trận (mảng 2 chiều) (hình 1). Với một ma trận đầu vào I và một kernel K kích thước cho trước, ta có:



Hình 1: Phép tích chập với kernel K kích thước 2 × 2 và input I kích thước 3 × 4

- Lần lượt trượt cửa sổ có kích thước bằng kernel theo chiều ngang, mỗi lần 1 đơn vị (chỉ số này tương ứng là stride), và thực hiện phép nhân từng phần tử của cửa sổ và K (piecewise multiplication), sau đó tổng các giá trị vừa tìm được, ta được giá trị của một pixel trong output.
- Khi trượt đến hết hàng, ta tăng số dòng lên 1 đơn vị, đặt cửa sổ lại đầu hàng và tiếp tục thực hiện phép nhân như bước trên.
- Việc tính toán sẽ dừng lại khi ta hoàn thành tính toán trên cửa sổ cuối cùng (dưới cùng, bên phải) của ma trận đầu vào I.

Tổng quát, với tensor đầu vào có kích thước W1 × H1 × D1, kernel kích thước K × K, số filter là F, bước nhảy stride S và lượng padding là P, thì qua lớp tích chập, ta sẽ nhận được

output là một tensor có kích thước W2 × H2 × D2, trong đó:

$$W_2 = \frac{W_1 - K + 2P}{S} + 1 \tag{3}$$

$$H_2 = \frac{H_2 - K + 2P}{S} + 1 \tag{4}$$

$$D_2 = F \tag{5}$$

2.2 Lớp gộp (Pooling layer)

Trong các mạng neuron tích chập, pooling chủ yếu được sử dụng để chỉnh sửa đầu ra nhận được của các lớp tích chập hay các hàm kích hoạt phi tuyến (nonlinear activation function) [6]. Về bản chất, hàm pooling thay thế đầu ra của một lớp bởi một thống kê của các đầu ra lân cận. Và, cũng nhờ cơ chế này, mà các mạng sử dụng phép tích chập có được một khả năng rất đáng được chú ý: tính bất biến với những phép dịch nhỏ . Điều này có nghĩa là, nếu ta dịch một bức ảnh đi vài pixels, đầu ra qua pooling sẽ hầu như không thay đổi. Ngoài ra, tính chất này đặc biệt hữu ích, nếu ta quan tâm rằng, liệu một đặc tính nào đó có xuất hiện trong đầu vào, hơn là chính xác đặc tính đó ở đâu.

Cũng bởi vì pooling tóm tắt thông tin của một vùng lân cận, nên trong thực tế, người ta thường sử dụng pooling với kích thước lân cận thường lớn hơn 1 đơn vị. Điều này cũng góp phần cải thiện khối lượng tính toán trong việc huấn luyện, bởi vì, qua mỗi lớp pooling, kích thước đầu ra sẽ có mỗi chiều có thể giảm đến k lần so với đầu vào gốc (ứng với kích thước k).

Trong thực tế, các giá trị thống kê thường được sử dụng để hiện thực pooling bao gồm giá trị trung bình (mean), giá trị lớn nhất (max).

2.3 Hàm kích hoạt phi tuyến (nonlinear)

2.3.1 Hàm ReLU

Hàm ReLU được định nghĩa như sau

$$f(x) = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Như vậy, có thể thấy, đối với các mạng sử dụng ReLU, một điểm mạnh lớn nhất mà mạng nhận được so với các hàm kích hoạt phi tuyến khác như sigmoid σ(x) hay hyperbolic tangent tanh(x) chính là về khối lượng tính toán. Cụ thể, thì so với các hàm phi tuyến khác, ReLU rất dễ để tính toán, vì ta chỉ đơn thuần so sánh giá trị đầu vào và 0. Bên cạnh đó, tại những điểm khả vi, ReLU có đạo hàm rất đơn giản hoặc là 0 hoặc 1, ứng với các giá trị âm và dương của đầu vào theo định nghĩa hàm ở phương trình trên. Chính việc tính toán dễ dàng đạo hàm cho phép việc thực hiện giải thuật back-propagation trở nên khả thi và hiệu quả .

Ngoài ra, ReLU cũng ngăn chặn việc mất gradient (vanishing gradient), vốn phổ biến khi sử dụng các hàm kích hoạt như σ(x). Hiện tượng này nghĩa là, khi giá trị của đầu vào x quá cao, đạo hàm của hàm sẽ tiến rất gần về 0, làm cho việc thực thi giải thuật back propagation trở nên khó khăn, do việc đạo hàm bằng 0 đồng nghĩa với việc tham số của mô hình sẽ không học được gì cả.

2.3.2 Hàm Sigmoid

Sigmoid là một hàm phi tuyến với đầu vào là các số thực và cho đầu ra nằm trong khoảng (0, 1) và được xem là xác xuất trong một số bài toán. Trong hàm sigmoid, một sự thay đổi nhỏ trong input dẫn đến một kết quả output không mấy thay đổi . Vì vậy, nó đem lại một đầu ra "mượt" hơn và liên tục hơn so với input. Định nghĩa hàm sigmoid và đạo hàm của nó:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{6}$$

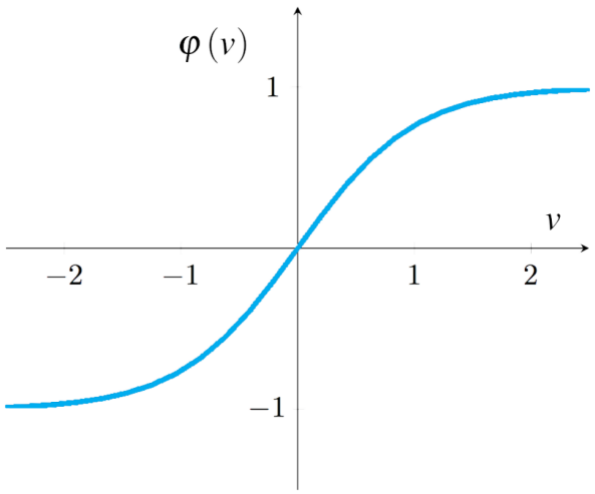
$$\frac{d}{dx}(\sigma(x)) = \sigma(x) \times (1-\sigma(x)) \tag{7}$$

Phương trình (6) và (7) cho chúng ta thấy rằng, hàm sigmoid là một hàm liên tục và đạo hàm của nó cũng khá đơn giản, dẫn đến việc áp dụng hàm vào mô hình mạng đem lại sự dễ dàng trong việc xây dựng mô hình và cập nhật tham số dựa trên back-propagation.

2.3.3 Hàm hyperbolic tangent

Hàm hyperbolic tangent là một hàm liên tục, có miền xác định là (-∞, +∞) và có miền giá trị là (-1, 1). Hàm này có công thức như sau:

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



Hình 2: Đồ thị hàm hyperbolic tangent

Khi sử dụng hàm sigmoid, nếu đầu vào có giá trị càng âm thì giá trị đầu ra của hàm sigmoid càng gần 0, điều đó sẽ làm cho quá trình huấn luyện mạng neuron trở nên chậm chạp, các trọng số được cập nhật với sự thay đổi giá trị rất ít. Trong trường hợp này, hàm hyperbolic tangent là một sự lựa chọn thay thế tốt cho hàm sigmoid.

2.4 Thang đo độ chính xác

Để phân loại cho một ảnh hay một pixel với nhãn tương ứng, người ta sử dụng bốn khái niệm đánh giá sự phân loại đúng hay sai một mẫu, ảnh nào đó. Đó là dương tính thật (True Positive - TP), dương tính giả (False Positive - FP), âm tính thật (True Negative - TN) và âm tính giả (False Negative - FN).

- Dương tính thật: Chỉ những ảnh, mẫu là đúng vật thể cần dự đoán và được mô hình dự đoán đúng.
- Dương tính giả: Chỉ những mẫu không phải là vật thể cần dự đoán nhưng được mô hình dự đoán, phân loại là đúng.
- Âm tính thật: Chỉ những mẫu không phải là vật thế cần dự đoán và được mô hình dự đoán là không phải.
- Âm tính giả: Chỉ những mẫu là vật thể cần dự đoán nhưng bị mô hình phân loại là không phải.

Để đánh giá một mô hình phân loại có chính xác hay không, người ta thường dùng 3 chỉ số phổ biến: độ chính xác (Accuracy), F-score và Sprarse categorical cross-entropy function.

2.4.1 Accuracy

Tiêu chí đầu tiên để đánh giá một mô hình phân loại có chính xác hay không, cũng là tiêu chí phổ biến nhất khi huấn luyện mạng neuron, đó chính là độ chính xác (Accuracy).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

Độ chính xác (Accuracy) được tính bằng tỉ số giữa số mẫu phân loại đúng trên tổng tất cả số mẫu đưa vào phân loại, có nghĩa là bằng tổng số mẫu âm tính thật và dương tính thật chia cho tổng của âm tính thật, dương tính thật, âm tính giả và dương tính giả như được trình bày trong công thức (8).

Trong phân vùng ảnh, dữ liệu được dự đoán là từng pixel của ảnh đầu ra. Như vậy, độ chính xác sẽ bằng tỉ số giữa số lượng pixel mà mô hình phân loại đúng trên tổng pixel của ảnh.

2.4.2 Precision và Recall

Với bài toán phân loại mà tập dữ liệu của các lớp là chênh lệch nhau rất nhiều, có một phép đo hiệu quả thường được sử dụng là Precision-Recall.

Precision được định nghĩa là tỉ lệ số điểm true positive trong số những điểm được phân loại là positive (TP + FP).

$$Precision = \frac{TP}{TP + FP} \tag{9}$$

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

$$Recall = \frac{TP}{TP + FN} \tag{10}$$

Precision cao đồng nghĩa với việc độ chính xác của các điểm tìm được là cao. Recall cao đồng nghĩa với việc True Positive Rate cao, tức tỉ lệ bỏ sót các điểm thực sự positive là thấp.

2.4.3 F-score

F-score (còn được gọi là F1 score hoặc F-measure) là một độ đo được sử dụng để đánh giá hiệu suất của một mô hình Học máy. Nó kết hợp độ chính xác (precision) và độ bao phủ (recall) thành một điểm số duy nhất.

Công thức tính F-score:

$$F - score = 2\frac{precision * recall}{precision + recall} \tag{11}$$

Độ chính xác trong việc đưa ra dự đoán tích cực được đo bằng độ bao phủ, trong khi xác định tất cả các trường hợp tích cực trong dữ liệu được đo bằng độ chính xác. F-score có giá trị từ 0 đến 1, với giá trị cao hơn cho thấy hiệu suất tốt hơn.

F-score trong Học máy thường được sử dụng khi mục tiêu là cân bằng độ chính xác với độ bao phủ và đặc biệt hữu ích khi lớp tích cực là hiếm gặp. Trong một hệ thống chẩn đoán y tế, có thể quan trọng hơn để đảm bảo mô hình có độ bao phủ cao (để giảm thiểu rủi ro bỏ sót các chẩn đoán), trong khi trong bộ lọc thư rác, đảm bảo độ chính xác cao là một ưu tiên hàng đầu (để giảm thiểu số lượng dương tính sai).

2.4.4 Sparse categorical cross-entropy

Sparse categorical cross-entropy (hay còn gọi là sparse softmax cross-entropy) là một hàm mất mát thường được sử dụng trong các bài toán phân loại đa lớp, đặc biệt là khi các nhãn (labels) được biểu diễn dưới dạng các giá trị nguyên không one-hot encoded.

Trong bài toán phân loại đa lớp, output của mô hình sẽ là một vector có độ dài bằng số lớp khác nhau trong bài toán. Mỗi phần tử trong vector đại diện cho xác suất mà mẫu dữ liệu đang xét thuộc về từng lớp. Để so sánh giữa output của mô hình và nhãn thực tế, chúng ta cần một hàm mất mát thích hợp.

Trong trường hợp sử dụng sparse categorical cross-entropy, nhãn thực tế được biểu diễn dưới dạng các giá trị nguyên (integer labels) từ 0 đến (số lớp - 1), chứ không phải one-hot encoded. Hàm mất mát này tính toán sai số giữa các phân phối xác suất dự đoán và nhãn thực tế.

Công thức của sparse categorical cross-entropy như sau:

$$Loss = -\sum_{i=1}^{outputsize} y * log\hat{y} \tag{12}$$

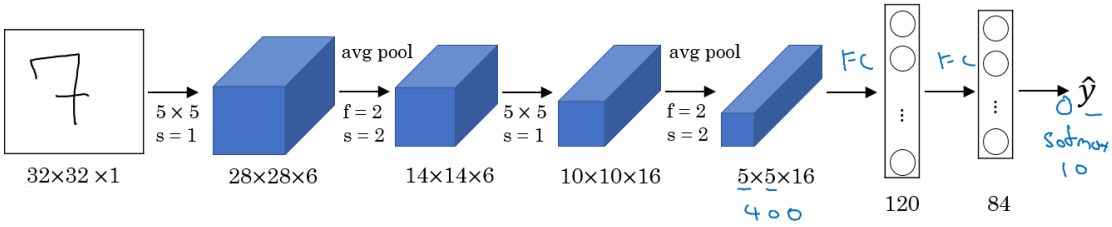
Trong đó:

- y là vector nhãn thực tế.
- $\hat{y}$ là vector phân phối xác suất dự đoán của mô hình.

Hàm mất mát này đo lường sự khác biệt giữa phân phối xác suất dự đoán (được tính bằng hàm softmax) và nhãn thực tế. Mục tiêu là tối thiểu hóa sai số giữa dự đoán và nhãn thực tế, từ đó cải thiện khả năng phân loại của mô hình.

2.5 Mạng học sâu LeNet-5

Mục tiêu của mô hình này là nhận dạng các chữ số viết tay trong một hình ảnh xám 32x32x1.



Hình 3: LeNet-5

Mô hình này đã được công bố vào năm 1998. Lớp cuối cùng không sử dụng softmax vào thời điểm đó. Mô hình này có 60.000 tham số.

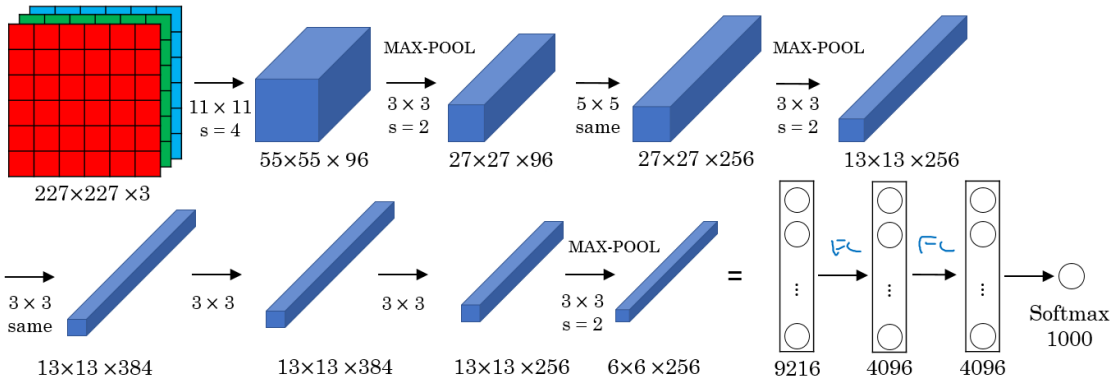
Kích thước của hình ảnh giảm dần khi số kênh tăng lên. Conv ==> Pool ==> Conv ==> Pool ==> FC ==> FC ==> softmax là một sắp xếp phổ biến.

Hàm kích hoạt được sử dụng trong bài báo là Sigmoid và Tanh. Trong hầu hết các trường hợp, các triển khai hiện đại sử dụng hàm kích hoạt RELU.

2.6 Mạng học sâu AlexNet

Mô hình này được đặt theo tên Alex Krizhevsky, người là tác giả chính của bài báo này. Các tác giả khác bao gồm Geoffrey Hinton.

Mục tiêu của mô hình là tham gia thử thách ImageNet, trong đó hình ảnh được phân loại vào 1000 lớp khác nhau. Đây là sơ đồ của mô hình:



Hình 4: AlexNet

Tóm tắt:

Conv => Max-pool => Conv => Max-pool => Conv => Conv => Conv => Max-pool ==> Flatten ==> FC ==> FC ==> Softmax

Tương tự như LeNet-5 nhưng lớn hơn.

Mô hình này có 60 triệu tham số so với 60 nghìn tham số của LeNet-5.

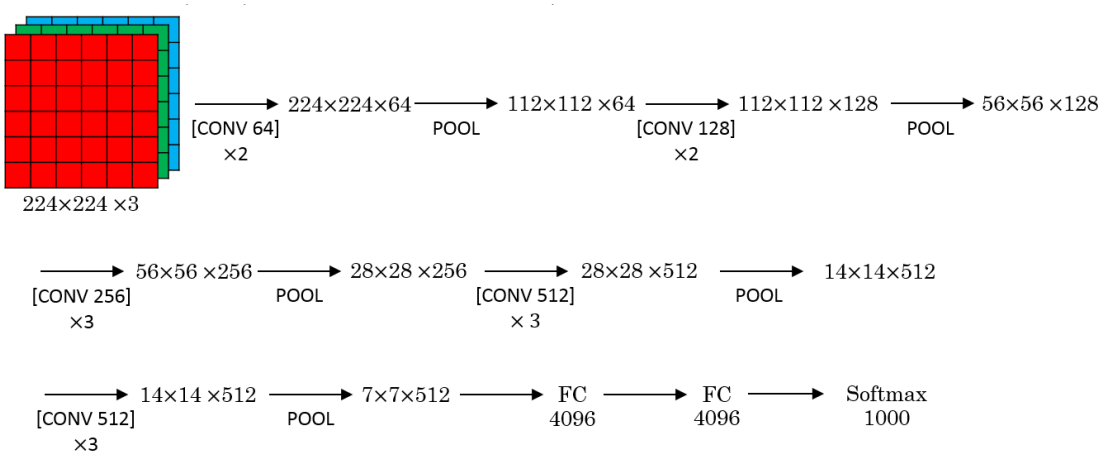
Mô hình sử dụng hàm kích hoạt RELU.

Bài báo gốc bao gồm việc sử dụng nhiều GPU và chuẩn hóa phản ứng cục bộ (RN).

Nhiều GPU đã được sử dụng vì GPU không nhanh như hiện nay. Các nhà nghiên cứu đã chứng minh rằng chuẩn hóa phản ứng cục bộ không giúp ích nhiều, vì vậy hiện tại không cần quan tâm đến hiểu hoặc triển khai nó. Mô hình này đã thuyết phục các nhà nghiên cứu về thị giác máy tính rằng học sâu (deep learning) rất quan trọng

2.7 Mạng học sâu VGG-16

Một phiên bản được chỉnh sửa cho AlexNet. Thay vì có nhiều siêu tham số, chúng ta sẽ có một mạng đơn giản hơn. Tập trung vào chỉ có những khối sau đây: CONV = bộ lọc 3 X 3, s = 1, same MAX-POOL = 2 X 2, s = 2 Dưới đây là kiến trúc:



Hình 5: VGG-16

Mạng này lớn ngay cả theo tiêu chuẩn hiện đại. Nó có khoảng 138 triệu tham số. Hầu hết các tham số nằm trong các lớp fully connected. Nó có tổng dung lượng bộ nhớ là 96MB cho mỗi hình ảnh chỉ cho quá trình truyền thẳng! Hầu hết bộ nhớ nằm trong các lớp đầu tiên. Số bộ lọc tăng từ 64 lên 128, rồi 256 và 512. Số 512 được lặp lại hai lần. Max-pooling là nguyên nhân duy nhất dẫn đến giảm kích thước. Có một phiên bản khác được gọi là VGG-19, đó là phiên bản lớn hơn. Nhưng hầu hết mọi người sử dụng VGG-16 thay vì VGG-19 vì chúng làm cùng một công việc. Bài báo về VGG hấp dẫn vì nó cố gắng đề ra một số quy tắc về việc sử dụng mạng neural tích chập (CNNs).

2.8 Mạng học sâu ResNets

Trong những năm gần đây, các mạng nơ ron, nhất là mạng nơ ron tích chập, đã được sử dụng rộng rãi trong những ứng dụng liên quan tới phân loại ảnh. Trong mạng nơ rơn tích chập, các lớp tích chập (convolutional layer) kết hợp với các lớp giảm chiều (pooling) có xu hướng làm cho kích thước của dữ liệu đầu vào bị thu nhỏ lại.

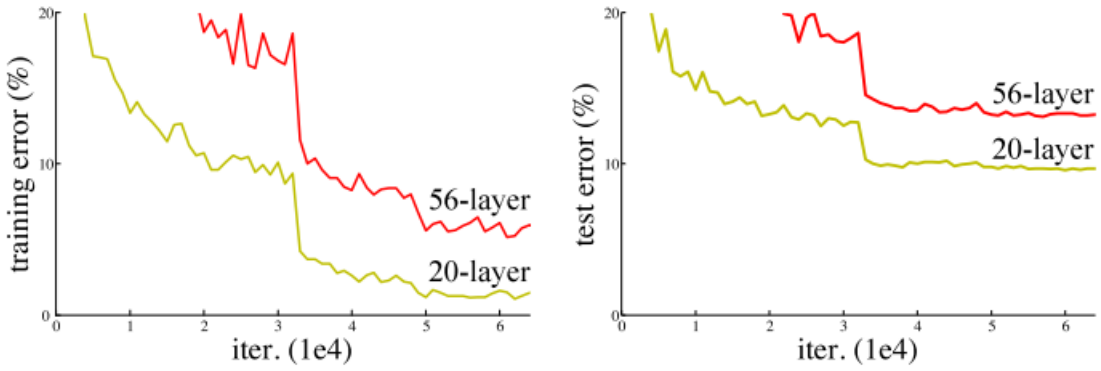
Trong bài toán phân vùng ảnh, đầu ra của mô hình là một ảnh khác, hay gọi là mặt nạ (mask) - phân chia chính xác từng vùng riêng biệt, tương ứng với đó là các vật thể trong ảnh, có chiều dài và rộng có thể bằng với ảnh đầu vào. Và để làm được điều này, một trong những cách đó chính là người ta xóa bỏ từ lớp mạng liên kết đầy đủ trở về sau, chỉ giữ lại phần tích chập và giảm chiều, đồng thời thêm vào cuối đó các lớp tích chập khác, và các lớp mở rộng, hay còn gọi

là upsampling 15. Cụ thể, từ các lớp tích chập và giảm chiều đầu vào, ta thu được một loạt các biểu đồ đặc tính (feature map) và từ những biểu đồ đặc tính ấy, thông qua việc mở rộng, tính toán với các lớp tích chập khác, ta sẽ có đầu ra là mặt nạ tương ứng.

Như vậy, mô hình có thể được tóm gọn lại với hai phần chính, phần mã hóa (encoder), với đầu ra là các biểu đồ đặc tính, và phần giải mã (decoder) chính là phần mở rộng để thu được mặt nạ phân vùng. Các mô hình như này có thể dễ dàng được cải tiến từ các mô hình mạng CNN vốn đã rất hiệu quả như VGG, ResNet, DenseNet, GoogLeNet,...

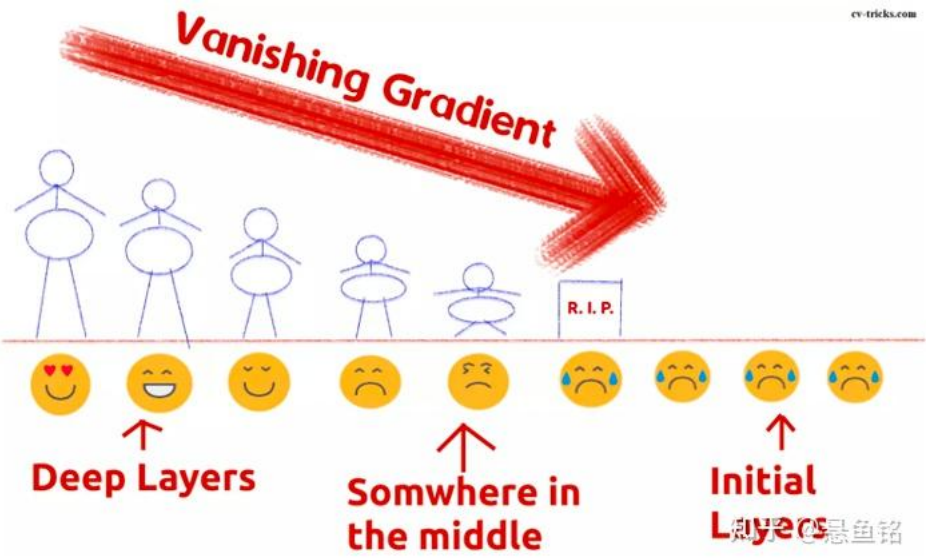
Mạng ResNet (R) là một mạng CNN được thiết kế để làm việc với hàng trăm lớp. Một vấn đề xảy ra khi xây dựng mạng CNN với nhiều lớp chập sẽ xảy ra hiện tượng Vanishing Gradient dẫn tới quá trình học tập không tốt.

Trước hết thì Backpropagation Algorithm là một kỹ thuật thường được sử dụng trong quá trình tranining. Ý tưởng chung của thuật toán lá sẽ đi từ output layer đến input layer và tính toán gradient của cost function tương ứng cho từng parameter (weight) của mạng. Gradient Descent sau đó được sử dụng để cập nhật các parameter đó.



Hình 6: Đồ thị training error và test error

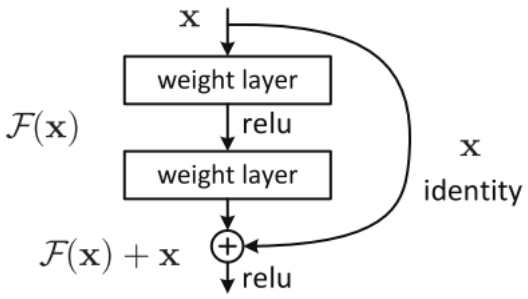
Toàn bộ quá trình trên sẽ được lặp đi lặp lại cho tới khi mà các parameter của network được hội tụ. Thông thường chúng ta sẽ có một hyperparametr (số Epoch - số lần mà traninig set được duyệt qua một lần và weights được cập nhật) định nghĩa cho số lượng vòng lặp để thực hiện quá trình này. Nếu số lượng vòng lặp quá nhỏ thì ta gặp phải trường hợp mạng có thể sẽ không cho ra kết quả tốt và ngược lại thời gian tranining sẽ lâu nếu số lượng vòng lặp quá lớn.



Hình 7: Vanishing Gradient

Tuy nhiên, trong thực tế Gradients thường sẽ có giá trị nhỏ dần khi đi xuống các layer thấp hơn. Dẫn đến kết quả là các cập nhật thực hiện bởi Gradients Descent không làm thay đổi nhiều weights của các layer đó và làm chúng không thể hội tụ và mạng sẽ không thu được kết quả tốt. Hiện tượng như vậy gọi là Vanishing Gradients.

Cho nên giải pháp mà ResNet đưa ra là sử dụng kết nối "tắt" đồng nhất để xuyên qua một hay nhiều lớp. Một khối như vậy được gọi là một Residual Block, như trong hình sau :



Hình 8: Residual Block

ResNet gần như tương tự với các mạng gồm có convolution, pooling, activation và fully-connected layer. Ảnh bên trên hiển thị khối dư được sử dụng trong mạng. Xuất hiện một mũi tên cong xuất phát từ đầu và kết thúc tại cuối khối dư. Hay nói cách khác là sẽ bổ sung Input

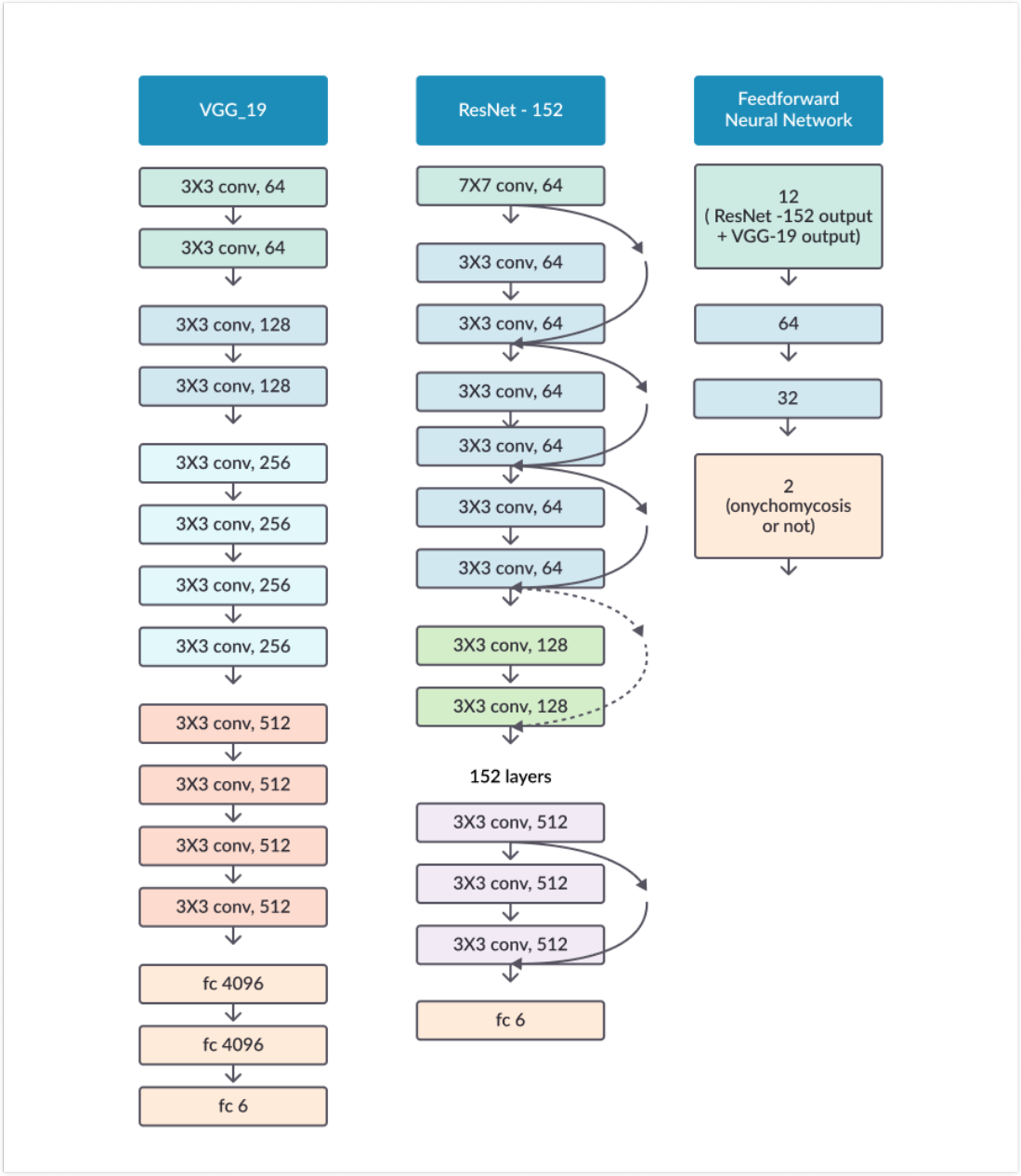
X vào đầu ra của layer, hay chính là phép cộng mà ta thấy trong hình minh họa, việc này sẽ chống lại việc đạo hàm bằng 0, do vẫn còn cộng thêm X. Với H(x) là giá trị dự đoán, F(x) là giá trị thật (nhãn), chúng ta muốn H(x) bằng hoặc xấp xỉ F(x). Việc F(x) có được từ x như sau:

$$X -> weight1 -> ReLU -> weight2 \tag{13}$$

Giá trị H(x) có được bằng cách:

$$F(x) + x -> ReLU \tag{14}$$

Như chúng ta đã biết việc tăng số lượng các lớp trong mạng làm giảm độ chính xác, nhưng muốn có một kiến trúc mạng sâu hơn có thể hoạt động tốt.



Hình 9: Kiến trúc mạng VGG-19 và ResMet-152

- Hình 1. VGG-19 là một mô hình CNN sử dụng kernel 3x3 trên toàn bộ mạng, VGG-19 cũng đã giành được ILSVRC năm 2014.

- Hình 2. ResNet sử dụng các kết nối tắt (kết nối trực tiếp đầu vào của lớp (n) với (n+x) được hiển thị dạng mũi tên cong. Qua mô hình nó chứng minh được có thể cải thiện hiệu suất trong quá trình training model khi mô hình có hơn 20 lớp.
- Hình 3. Tổng cộng có 12 đầu ra từ ResNet-152 và VGG-19 đã được sử dụng làm đầu vào cho mạng có 2 lớp hidden. Đầu ra cuối cùng được tính toán thông qua hai lớp ẩn (hidden). Việc xếp chồng các lớp sẽ không làm giảm hiệu suất mạng.Với kiến trúc này các lớp phía trên có được thông tin trực tiếp hơn từ các lớp dưới nên sẽ điều chỉnh trọng số hiệu quả hơn.

2.9 Mạng Tích chập Kết nối Dày đặc (DenseNet)

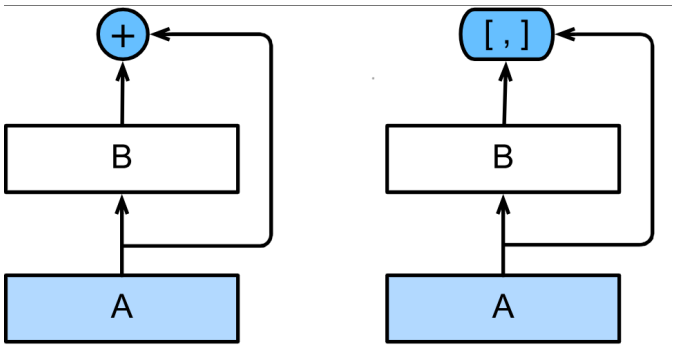
ResNet đã làm thay đổi đáng kể quan điểm về cách tham số hóa các hàm số trong mạng nơ-ron sâu. Ở một mức độ nào đó, DenseNet có thể được coi là phiên bản mở rộng hợp lý của ResNet. Để hiểu cách đi đến kết luận đó, ta cần tìm hiểu một chút lý thuyết. Nhắc lại công thức khai triển Taylor cho hàm một biến vô hướng như sau

$$f(x) = f(0) + f'(x)x + 12f''(x)x^2 + 16f'''(x)x^3 + o(x^3) \tag{15}$$

Điểm mấu chốt là khai triển Taylor phân tách hàm số thành các số hạng có bậc tăng dần. Tương tự, ResNet phân tách các hàm số thành

$$f(x) = x + g(x) \tag{16}$$

Cụ thể, ResNet tách hàm số f thành một số hạng tuyến tính đơn giản và một số hạng phi tuyến phức tạp hơn. Nếu ta muốn tách ra thành nhiều hơn hai số hạng thì sao? Một giải pháp đã được đề xuất bởi [Huang et al., 2017] trong kiến trúc DenseNet. Kiến trúc này đạt được hiệu suất kỉ lục trên tập dữ liệu ImageNet.

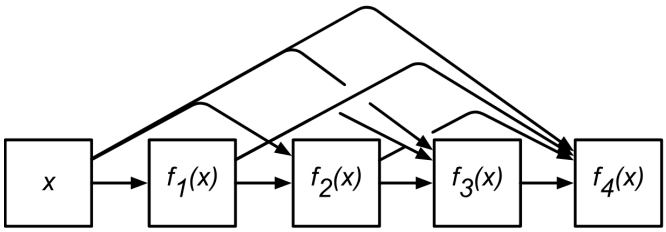


Hình 10: Sự khác biệt chính giữa ResNet (bên trái) và DenseNet (bên phải) trong các kết nối xuyên tầng: sử dụng phép cộng và sử dụng phép nối.

Như được thể hiện trong hình 10, điểm khác biệt chính là DenseNet nối đầu ra lại với nhau thay vì cộng lại như ở ResNet. Kết quả là ta thực hiện một ánh xạ từ x đến các giá trị của nó sau khi áp dụng một chuỗi các hàm với độ phức tạp tăng dần.

$$x- > [x, f1(x), f2(x, f1(x)), f3(x, f1(x), f2(x, f1(x)), \dots]. \tag{17}$$

Cuối cùng, tất cả các hàm số này sẽ được kết hợp trong một Perceptron đa tầng để giảm số lượng đặc trưng một lần nữa. Lập trình thay đổi này khá đơn giản — thay vì cộng các số hạng với nhau, ta sẽ nối chúng lại. Cái tên DenseNet phát sinh từ việc đồ thị phụ thuộc giữa các biến trở nên khá dày đặc. Tầng cuối cùng của một chuỗi như vậy được kết nối "dày đặc" tới tất cả các tầng trước đó. Thành phần chính của DenseNet là các khối dày đặc và các tầng chuyển tiếp. Các khối dày đặc định nghĩa cách các đầu vào và đầu ra được nối với nhau, trong khi các tầng chuyển tiếp kiểm soát số lượng kênh sao cho nó không quá lớn. Các kết nối dày đặc được biểu diễn trong hình 11.



Hình 11: Các kết nối dày đặc trong DenseNet

V PHƯƠNG PHÁP NGHIÊN CỨU

1 Giới thiệu về bộ dữ liệu

Trong bài nghiên cứu này, nhóm tác giả sử dụng tập dữ liệu IQ-OTH/NCCD - Lung Cancer làm tập dữ liệu mẫu để huấn luyện mô hình. Đây là tập ảnh CT chụp các tế bào não. Một số đặc điểm của Brain Tumours được mô tả như sau:

- Trang web chứa tập dữ liệu: DATASET
- Loại hình ảnh: CT Scan
- Màu sắc: trắng đen
- Số lượng ảnh: 1190

```
{'begin': {'512 x 512' : 120},
'malignants' : {'512 x 512':501,
'404 x 511': 1,
'512 x 801' :28,
'512 x 623' :31,
'normal' : {'512 x 512' : 415, '331 x 506' :1}}
```

Bộ dữ liệu ảnh bao gồm các hình ảnh với kích thước khác nhau. Kích thước chính của hình ảnh là 512 x 512 pixel. Trong đó có 120 hình ảnh begin, có 501 hình ảnh được phân loại là "malignants" (ác tính) và có kích thước 512 x 512 pixel.

Ngoài ra, có một số hình ảnh "malignants" khác với kích thước khác như sau:

- 1 hình ảnh có kích thước 404 x 511 pixel.
- 28 hình ảnh có kích thước 512 x 801 pixel.
- 31 hình ảnh có kích thước 512 x 623 pixel.

Bên cạnh đó, trong bộ dữ liệu còn có hình ảnh được phân loại là "normal" (bình thường). Có 415 hình ảnh "normal" có kích thước 512 x 512 pixel và 1 hình ảnh có kích thước 331 x 506 pixel.

Nhận xét: Với bộ dữ liệu hình ảnh được mô tả như trên, có một số điểm ảnh hưởng đến độ hiệu quả của các mô hình như ResNet, VGG hay LeNet-5:

- Số lượng mẫu: Mặc dù tổng số ảnh khá lớn (>1000 ảnh) nhưng một số loại ảnh chỉ có số lượng ít (<30 ảnh). Điều này có thể gây ra vấn đề thiếu dữ liệu đối với mô hình khi huấn luyện.
- Độ phân phối mẫu: Các ảnh có kích thước và định dạng khác nhau (512x512, 404x511, 331x506,...). Điều này làm giảm tính đồng nhất của dữ liệu, ảnh hưởng đến khả năng học của mô hình.
- Một số loại ảnh chỉ có một vài mẫu (ví dụ chỉ 1 ảnh 404x511), gây khó khăn cho việc phân loại.

2 Thu thập và làm sạch dữ liệu

Đầu tiên, nhóm sẽ resize ảnh lại thành kích thước 256 x 256 pixel.

```
img_size = 256
for i in categories:
cnt, samples = 0, 3
fig, ax = plt.subplots(samples, 3, figsize=(15, 15))
fig.suptitle(1)

path = os.path.join(directory, i)
class_num = categories.index(i)
for curr_cnt, file in enumerate(os.listdir(path)):
filepath = os.path.join(path, file)
img = cv2.imread(filepath, 9)

img0 = cv2.resize(img, (img_size, img_size))

img1 = cv2.GaussianBlur(img0, (5, 5), 0)

ax[cnt, 0].imshow(img)
ax[cnt, 1].imshow(img0)
ax[cnt, 2].imshow(img1)
cnt += 1
if cnt == samples:
break

plt.show()
```

Tiền xử lí dữ liệu:

```
data_dir = 'Data/data/The IQ-OTHNCCD lung cancer dataset/'

categories = ['bengin', 'malignant', 'normal']

def preprocess_image(file_path):
    img = cv2.imread(file_path, 0)
    img = cv2.resize(img, (224, 224))
    img = img / 255.0 # Chuan hoa pixel trong khoang [0, 1]
    return img

data = []
labels = []

for category in categories:
    path = os.path.join(data_dir, category)
    label = categories.index(category)
    for file in os.listdir(path):
        file_path = os.path.join(path, file)
        img = preprocess_image(file_path)
        data.append(img)
        labels.append(label)

data, labels = shuffle(data, labels, random_state=42)

# Kiem tra can bang du lieu
print('Data length:', len(data))
print('labels counts:', Counter(labels))

# normalize
X = np.array(data).reshape(-1, 224, 224, 1)
y = np.array(labels)
```

Kết quả:

```
Data length: 1097
labels counts: Counter({1: 561, 2: 416, 0: 120})
```

Split data

```
X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.3,
    random_state=10)

print('Train length:', len(X_train), X_train.shape)
print('Test length:', len(X_valid), X_valid.shape)
```

```
Train length: 822 (822, 224, 224, 1)
Test length: 275 (275, 224, 224, 1)
```

Data Preparation

```
train_datagen = ImageDataGenerator()
val_datagen = ImageDataGenerator()
train_generator = train_datagen.flow(X_train, y_train, batch_size=16, shuffle=True)
validation_generator = val_datagen.flow(X_valid, y_valid, batch_size=16,
    shuffle=True)
```

3 Xây dựng và huấn luyện mô hình

Sau khi thu thập và tiền xử lý dữ liệu, nhóm tác giả hiện sẽ đưa dữ liệu vào mô hình để huấn luyện. Các layer đã được nhóm tác giả đón nhận để xây dựng mô hình từ Keras bao gồm Conv2D, MaxPooling2D, Dropout, Dense, Activation, Flatten, concatenate, BatchNormalization và Conv2DTranspose. Bên cạnh đó sẽ sử dụng thêm các applications có sẵn như VGG16, Resnet50 để tiến hành xây dựng mô hình. Sau đó sẽ đưa vào các mô hình VGG16, AlexNet, LeNet-5, ResNet50 để tiến hành huấn luyện.

3.1 Xây dựng mô hình

```
resnet_base = ResNet50(weights=None, include_top=False,
    input_shape=X_train.shape[1:])

# Freeze the pre-trained layers so they are not updated during training
for layer in resnet_base.layers:
    layer.trainable = False

# Create a new model
model_resnet = Sequential()

# Add the ResNet50 base model
model_resnet.add(resnet_base)

# Added custom classification layers on top
model_resnet.add(Flatten())
model_resnet.add(Dense(128, activation='relu'))
model_resnet.add(Dense(3, activation='softmax'))

# Print the summary of the model
model_resnet.summary()
```

3.2 Huấn luyện mô hình:

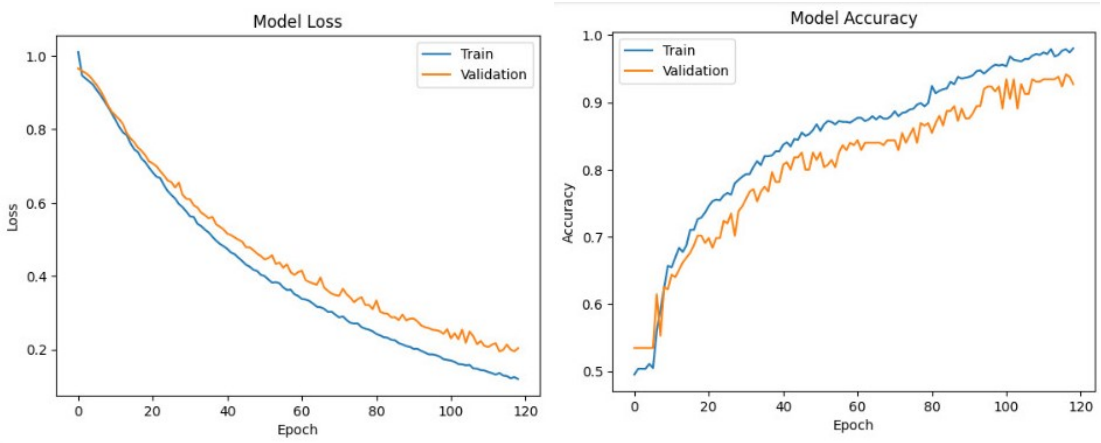
```
# Create an EarlyStopping callback to monitor validation loss and stop training
    if it doesn't improve for 5 consecutive epochs.
callback = EarlyStopping(monitor='val_loss', patience=5)
#compile model
model_resnet.compile(loss='sparse_categorical_crossentropy', optimizer='adam',
    metrics=['accuracy'])
#Training model
history = model_resnet.fit(train_generator, epochs=160,
    validation_data=validation_generator, callbacks=[callback])
```

VI KẾT QUẢ VÀ ĐÁNH GIÁ

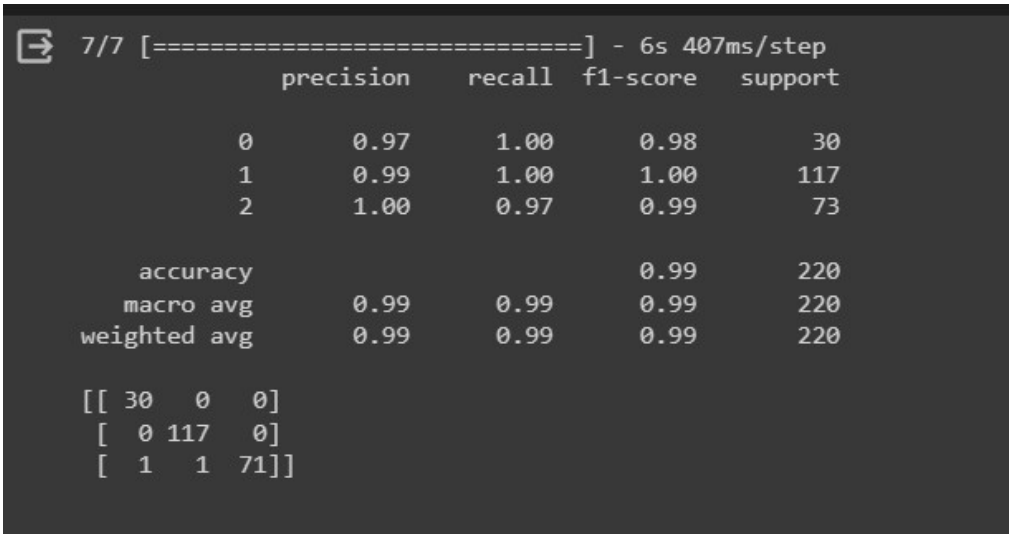
1 Kết quả đạt được

1.1 Mô hình ResNet

Sau 20 epochs huấn luyện, tương đương khoảng 1 giờ huấn luyện trên thiết bị cá nhân, nhóm tác giả thu được kết quả của hàm mất mát và độ chính xác như sau:



Hình 12: Training và validation loss-accuracy qua từng epoch của ResNet



Hình 13: Kết quả huấn luyện ResNet

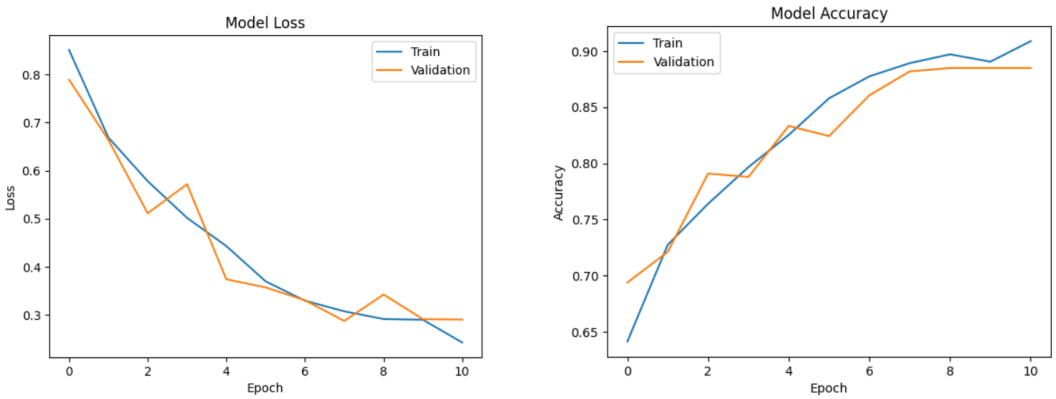
Có thể thấy, việc học của mô hình ResNet tương đối ổn định, thể hiện ở việc training loss giảm dần qua mỗi epoch, và giảm còn 0.0726 ở epoch cuối cùng. Đây là một dấu hiệu rất tốt cho thấy được sự thành công của mô hình.

Khi xét đến thước đo về hiệu năng, có thể thấy độ chính xác của mô hình nhất quán với đồ thị của hàm loss. Càng đến những epoch sau, việc học của mô hình càng tốt hơn và cho được một kết quả rất tốt với chỉ số accuracy sau vòng huấn luyện thứ 20 là 0.9818, hay một cách tương đương, vào 98.18%.

Nhìn vào 2 biểu đồ, sự chênh lệch giá trị training loss và validation loss, cũng như training accuracy và validation accuracy khá thấp, chỉ khoảng 0.1. Điều này cho thấy tính hiệu quả của mô hình khi chạy trên tập dữ liệu mới, tránh được trường hợp **overfitting**.

Như vậy, mô hình ResNet đã thể hiện một hiệu suất ấn tượng. Đây là một minh chứng rõ ràng về sự thành công về khả năng phân loại xuất sắc của mô hình ResNet.

1.2 LetNet-5

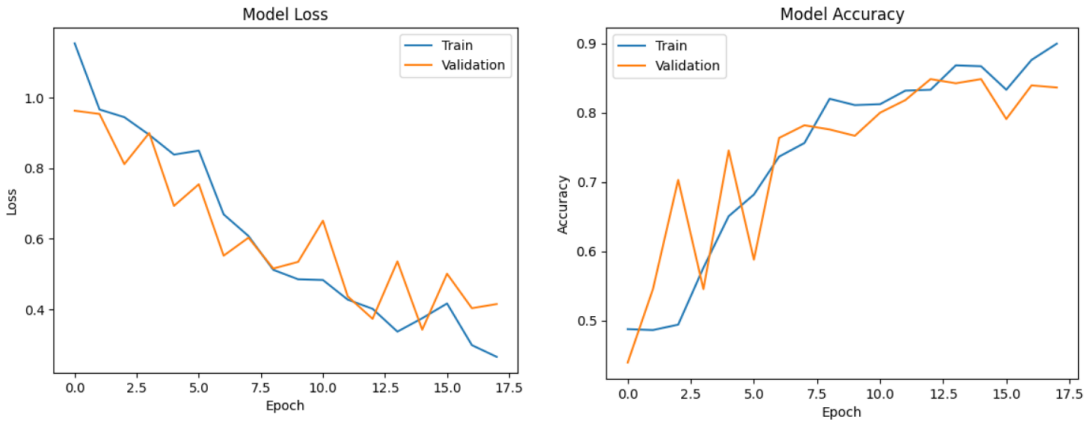


Hình 14: Training và validation loss-accuracy qua từng epoch của LetNet 5

```
7/7 [==============================] - 0s 25ms/step
              precision    recall  f1-score   support

           0       0.94      0.97      0.95        30
           1       0.99      0.97      0.98       117
           2       0.95      0.96      0.95        73

    accuracy                           0.97       220
   macro avg       0.96      0.97      0.96       220
weighted avg       0.97      0.97      0.97       220

[[ 29   0   1]
 [  0 114   3]
 [  2   1  70]]
```

Hình 15: Kết quả huấn luyện LetNet-5

1.3 AlexNet

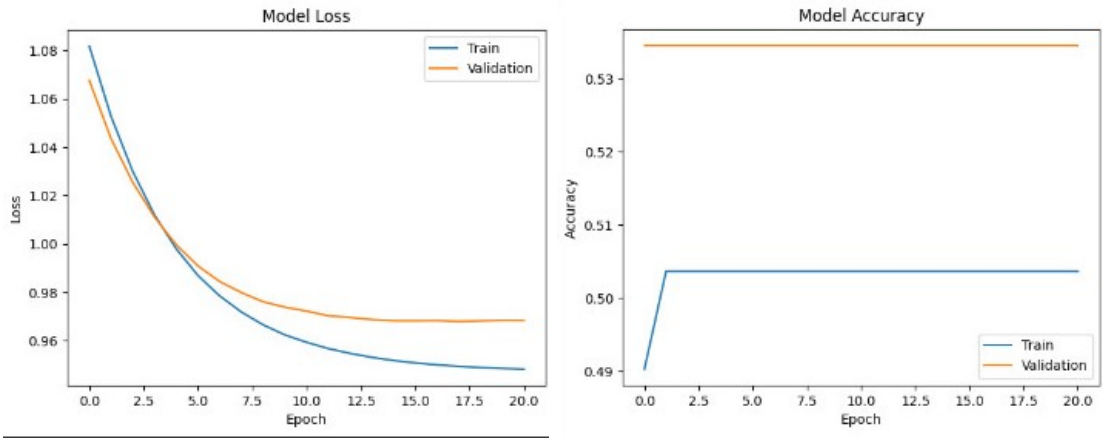


Hình 16: Training và validation loss-accuracy qua từng epoch của AlexNet

```
7/7 [==============================] - 1s 100ms/step
              precision    recall  f1-score   support

           0       0.14      1.00      0.24        30
           1       0.00      0.00      0.00       117
           2       0.00      0.00      0.00        73

    accuracy                           0.14       220
   macro avg       0.05      0.33      0.08       220
weighted avg       0.02      0.14      0.03       220
```

Hình 17: Kết quả huấn luyện AlexNet

1.4 VGG-16

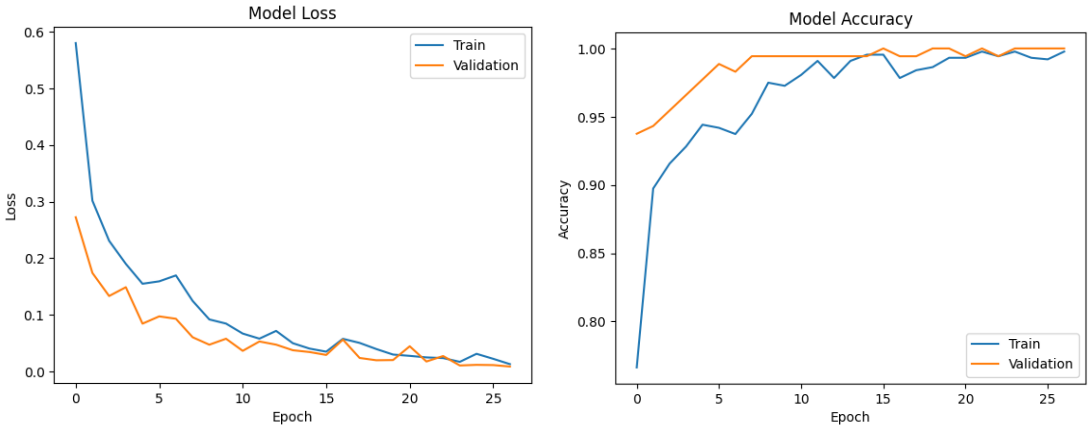


Hình 18: Training và validation loss-accuracy qua từng epoch của VGG-16

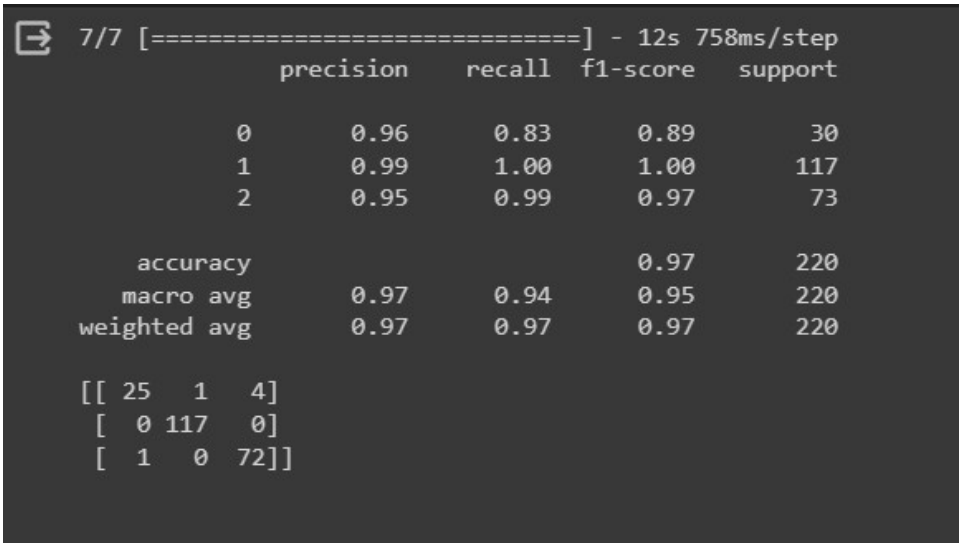
```
7/7 [==============================] - 14s 1s/step
              precision    recall  f1-score   support

           0       1.00      0.83      0.91        30
           1       0.99      1.00      1.00       117
           2       0.95      1.00      0.97        73

    accuracy                           0.98       220
   macro avg       0.98      0.94      0.96       220
weighted avg       0.98      0.98      0.98       220

[[ 25   1   4]
 [  0 117   0]
 [  0   0  73]]
```

Hình 19: Kết quả huấn luyện VGG

1.5 DenseNet121



Hình 20: Training và validation loss-accuracy qua từng epoch của DenseNet121



Hình 21: Kết quả huấn luyện DenseNet

Có thể thấy, việc học của các mô hình khác không được ổn định như ResNet, thể hiện ở những chỗ training loss tăng đột biến. Như ở mô hình LetNet5 training loss tăng ở epoch số 3 và 8; ở mô hình AlexNet training loss tăng ở nhiều epoch khác nhau như 3,5,7,..; còn ở mô hình VGG-16 training loss giảm tương đối tốt nhưng độ chính xác của mô hình lại rất hạn chế, chỉ ở mức 0.525.

Như vậy, khi so sánh việc học của ResNet và 3 mô hình khác gồm LetNet-5, AlexNet và VGG-16, mô hình ResNet đã thể hiện tính phân loại rất hiệu quả trên tập dữ liệu cần học. Trong khi ResNet thể hiện sự ổn định với mức độ giảm training loss qua mỗi epoch và đạt được

một độ chính xác ấn tượng, các mô hình khác lại gặp phải các khó khăn riêng của chúng.

2 So sánh kết quả đạt được của các mô hình

Dựa vào kết quả sau khi huấn luyện các mô hình, nhóm tác giả có bảng so sánh như nhau:

	Begin			Malignants			Normal		
Model	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
LetNet-5	0.94	0.97	0.95	0.99	0.97	0.98	0.95	0.96	0.95
AlexNet	0.14	1.00	0.24	0.00	0.00	0.00	0.00	0.00	0.00
VGG-16	1.00	0.83	0.91	0.99	1.00	1.00	0.95	1.00	0.97
DenseNet	0.96	0.83	0.89	0.99	1.00	1.00	0.95	0.99	0.97
ResNet	**1.00**	**0.87**	**0.93**	**1.00**	**0.99**	**1.00**	**0.95**	**1.00**	**0.98**

Nhìn vào bảng trên, có thể thấy rằng mô hình ResNet cho kết quả tốt nhất trong việc phân loại các ảnh CT của phổi. Mô hình này có kết quả precision, recall, F1-score cao nhất cho cả ba lớp: Begin, Malignants và Normal. Điều này cho thấy, trong bài toán phân loại này, mô hình ResNet có khả năng học được các đặc trưng quan trọng của các khối u và phân biệt nó hiệu quả hơn 3 loại mô hình LetNet-5, AlexNet, VGG-16.

VII KẾT LUẬN, KHUYẾN NGHỊ VÀ ĐỀ XUẤT

1 Những mặt đã làm được

Nhìn chung, so với những mục tiêu đề ra ban đầu, nhóm tác giả đã hoàn thiện được hầu hết các mục tiêu đề ra. Cụ thể:

- Nhóm tác giả đã có được những kiến thức tương đối vững vàng về các mạng học sâu dựa trên CNN và các vấn đề liên quan, bao gồm các lớp trong mạng học sâu, các loại hàm kích hoạt phi tuyến, các thang đo độ tốt của mô hình và một số ý tưởng cơ bản nhất trong cơ chế vận hành của các mạng học sâu.
- Nhóm tác giả cũng đã làm việc được trên một tập dữ liệu là tập ảnh CT, chụp phần phổi của cơ thể người, và có thể khai thác được khối CT ban đầu để trích xuất ra các lát ảnh 2D phục vụ cho việc phân đoạn ảnh tìm khối u.
- Nhóm tác giả cũng đã hiện thực lại được các mạng học sâu, để phân vùng ảnh đã được trích xuất từ trước, trong đó, tăng cường dữ liệu được sử dụng để khắc phục hạn chế về số lượng ảnh có trong tập dữ liệu. Mô hình cũng đã được kiểm chứng về độ tốt thông qua một thang đo độ chính xác phù hợp với đặc điểm đầu vào, là precision, recall và F-score, và mô hình cũng đã thật sự có thể phân vùng được khối u, tuy không phải với độ chính xác dẫn đầu ở thời điểm hiện tại, nhưng là một ngưỡng có thể chấp nhận được so với thời gian mà nhóm tác giả có được để thực hiện nghiên cứu khoa học lần này.
- Ngoài ra, bên cạnh những kết quả về mặt chuyên môn, nhóm tác giả cũng có thêm rất nhiều kinh nghiệm quý báu để có thể làm việc nhóm, cũng như phong cách làm việc cá nhân, đặc biệt là việc phân bổ thời gian và nguồn lực hợp lý.

2 Những hạn chế cần khắc phục và những định hướng tiếp theo

Bên cạnh những mặt đã làm được, tính đến thời điểm hiện tại, nhóm tác giả vẫn còn tồn đọng một số vấn đề, gây ra bởi sự hạn chế về mặt thời gian, tài nguyên có sẵn cũng như kiến thức của nhóm tác giả. Một cách chi tiết,

- Việc huấn luyện mô hình của nhóm tác giả đã diễn ra trên một tập dữ liệu đã được thu nhỏ đi, chứ không phải dữ liệu gốc ban đầu, do tài nguyên tính toán mà nhóm tác giả có sẵn tương đối hạn chế.
- Thực tế ngày nay, yêu cầu của các chuyên gia đối với bài toán phân vùng cho khối u, là một mô hình có thể phân vùng tốt ngay trên khối CT hay MRI ban đầu, chứ không phải là các lát ảnh 2D như nhóm tác giả đã trích xuất. Song, với thời lượng hạn chế của nghiên cứu khoa học cũng như đây là lần đầu tiên nhóm tác giả tiếp cận với đề tài, nhóm tác giả chưa đủ khả năng để có thể đáp ứng yêu cầu trên, mà chỉ có thể hiện thực lại một trường hợp đơn giản hơn của yêu cầu gốc ban đầu.

Và cũng chính vì vậy, mà ở giai đoạn tiếp theo, nhóm tác giả sẽ tìm cách cải thiện hơn nữa hiệu năng của mô hình đã xây dựng, đồng thời, nghiên cứu và tìm tòi cách xây dựng một mạng học sâu có thể phân vùng được ngay trên ảnh CT hay MRI. Và khi đó, nhóm tác giả sẽ so sánh kết quả đạt được so với các mô hình khác cùng được dùng để giải quyết vấn đề này, với hy vọng có thể xây dựng được một mô hình đủ tốt để các chuyên gia có thể sử dụng như là một bản tham chiếu cho các quyết định chẩn đoán của mình, góp phần phục vụ tốt hơn cho các bệnh nhân.

VIII Tài liệu tham khảo

[1] World Health Organization (WHO)- Estimated number of new cancer cases, 2023.

[2] D. Forsyth and J.Ponce - Computer vision: a modern approach. Prentice Hall Professional Technical Reference, 2002.

[3] Ian Goodfellow, Yoshua Bengio, Aaron Courville - Deep Learning (Adaptive Computation and Machine Learning series).

[4] Institute of Electrical and Electronics Engineers - Proceedings of the IEEE, 1998.

[5] Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton - ImageNet Classification with Deep Convolutional Neural Networks, 2012

[6] Karen Simonyan and Andrew Zisserman - Very Deep Convolutional Networks for Large-Scale Image Recognition, 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun - Deep Residual Learning for Image Recognition. 2015.