ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN - ĐIỆN TỬ
BỘ MÔN VIỄN THÔNG

ĐỀ CƯƠNG LUẬN VĂN TỐT NGHIỆP

MÔ HÌNH DỰ ĐOÁN BỆNH XƠ PHỔI SỬ DỤNG MÁY HỌC

SINH VIÊN THỰC HIỆN:
NGUYỄN MỸ HẰNG-1711215
NGUYỄN CAO HOÀNG NAM-1712241

GIẢNG VIÊN HƯỚNG DẪN:
PGS. TS. HÀ HOÀNG KHA

Thành phố Hồ Chí Minh, tháng 12 năm 2020

Mục lục

Danh sách hình vẽ

Lời cảm ơn

Trong thời gian thực hiện đề tài này,chúng em đã nhận được sự định hướng, giúp đỡ tận tình của giáo viên hướng dẫn và những ý kiến đóng góp từ bạn bè. Nhờ những sự giúp đỡ đó, nhóm đã hoàn thành được một phần đề tài và viết ra định hướng để phát triển đề tài cho Luận văn tốt nghiệp sắp tới.hiện

Nhóm xin gửi lời cảm ơn sâu sắc đến thầy Hà Hoàng Kha, giảng viên hướng dẫn trực tiếp nhóm từ đồ án đến thực tập tốt nghiệp và đề cương luận văn tốt nghiệp. Sự nhiệt tình của thầy đã tiếp thêm cho nhóm nhiều động lực. Nhóm cũng hi vọng đối với đề cương và luận văn sắp tới, em có thể tiếp tục học hỏi nhiều điều hơn từ thầy.

Cuối cùng, đề tài hiện tại vẫn còn nhiều điều thiếu sót, rất mong nhận được những ý kiến đóng góp của thầy cô và các bạn để nhóm có thể hoàn thiện hơn đề tài này.

Sinh viên thực hiện
Nguyễn Mỹ Hằng
Nguyễn Cao Hoàng Nam

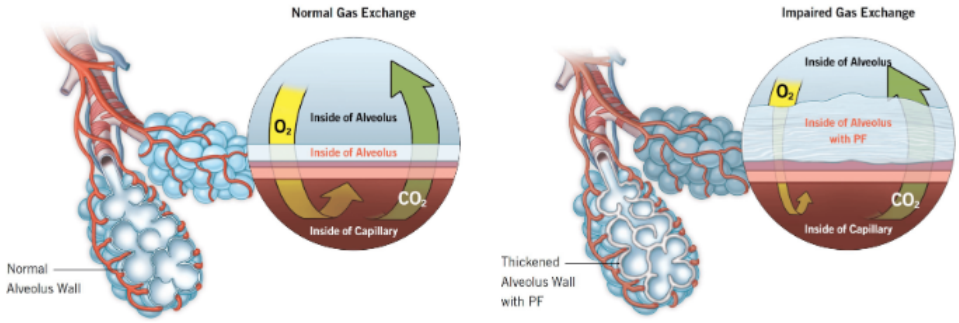
Chương 1

GIỚI THIỆU ĐỀ TÀI

1.1 Tổng quan về bệnh xơ phổi

1.1.1 Bệnh xơ phổi là gì

Xơ phổi là bệnh gây sẹo ở phổi và xơ vôi phổi. Xơ có thể phá hủy phổi bình thường và khiến Oxy khó đi vào máu. Khi cơ thể thiếu Oxy, các hoạt động khác bị ảnh hưởng nghiêm trọng, có thể đe dọa đến tính mạng người bệnh. Xơ phổi khiến mô ở sâu trong phổi bị tổn thương, dày và cứng hơn do mất tính đàn hồi và tạo sẹo, lâu dài làm biến chứng suy tim. Sự khác biệt giữa phổi bình thường và phổi bị xơ được mô tả ở hình 1.1. Ở giai đoạn cuối, nồng độ oxy trong máu giảm xuống quá thấp dẫn đến rối loạn nhịp tim, suy hô hấp trên nền cạn, có thể dẫn đến tử vong.



Hình 1.1: So sánh phổi bình thường và phổi bị xơ

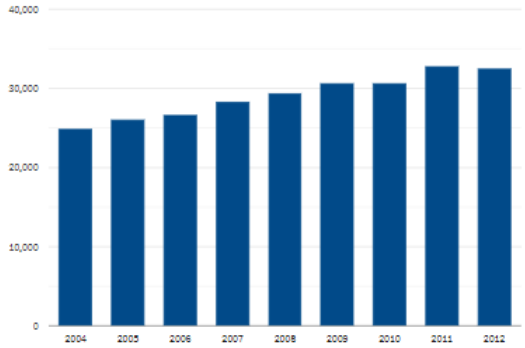
Trên thế giới, bệnh xơ phổi Bệnh xơ phổi có thể gây ra một số triệu chứng như: Mệt mỏi, đau nhức cơ khớp, sụt cân, ho khan, khó thở. Trong đó, khó thở là triệu chứng điển hình của bệnh xơ phổi. Khi bệnh nhân xuất hiện triệu chứng này tức là bệnh đã ở giai đoạn nặng và không thể phục hồi được tình trạng tổn thương của phổi, mặc dù có thể triệu chứng đã thuyên giảm. Cuối cùng, việc khó thở trở nên nghiêm trọng hơn, ngay cả trong hoạt động sinh hoạt thường ngày [2].

Xơ phổi là bệnh thường gặp ở bệnh nhân trong khoảng $50-70$ tuổi. Tiên lượng của bệnh chỉ trong $3-5$ năm. Hiện nay vẫn chưa xác định chính xác nguyên

nhân nào gây bệnh xơ phổi, tuy nhiên có một số tác nhân được cho là tăng khả năng mắc bệnh như: Hút thuốc, sống và làm việc trong môi trường bị ô nhiễm, đang điều tri và sử dụng một số loại thuốc.

1.1.2 Tình trạng bệnh xơ phổi trên thế giới và Việt Nam

Theo thống kê của British Lung Foundation,năm 2012 ở Vương quốc Anh có khoảng 32,500 người mắc xơ phổi trên tổng dân số là 63 triệu dân, tức là cứ 100.000 người thì có khoảng 50 người mắc bệnh xơ phổi. Con số này đã tăng đáng kể từ 37 người vào năm 2004. Như hình 1.2, số lượng người mắc xơ phổi có xu hướng tăng dần theo thời gian do môi trường ngày càng ô nhiễm.



Hình 1.2: Thống kê số người mắc xơ phổi tại Anh giai đoạn 2004-2012 [1].

Tương tự, những năm trở lại đây tình trạng ô nhiễm không khí nước ta ngày càng nghiêm trọng, nhất là ở những thành phố lớn lượng khí thải do phương tiện giao thông và nhà máy xí nghiệp thải ra môi trường với cường độ cao. Đây là một trong những nguyên nhân làm cho tỷ lệ người mắc bệnh lý liên quan đến phổi ngày càng tăng cao.

1.1.3 Chẩn đoán và điều trị xơ phổi

Hiện nay, bệnh xơ phổi được chẩn đoán thông qua các phương pháp như Chụp CT (Computed Tomography) vùng ngực, kiểm tra chức năng hô hấp. Sau khi dự đoán bệnh, các biện pháp điều trị chủ yếu là dùng thuốc để giảm thiểu, cải thiện các triệu chứng bệnh, làm chậm đi quá trình phát triển bệnh.

Chụp CT là sử dụng máy tính và máy X-quang để tạo ra ảnh cắt ngang của phổi, giúp nhận dạng các tổn thương ở phổi một cách chi tiết, rõ ràng như kích thước, mức độ tổn thương. Chụp CT sử dụng năng lượng tia X do đó việc chụp nhiều lần sẽ khiến bệnh nhân nhiễm xạ.

Đo thông số FVC (Forced Vital Capacity) là đo dung tích phổi (ml) hay lượng khí mà cơ thể có thể thở ra khi nỗ lực thở ra tối đa.

Xơ phổi là một loại bệnh tiến triển theo thời gian.Việc phát hiện bệnh sớm giúp việc điều trị dễ dàng hơn. Việc chụp CT tại một thời điểm không thể cho biết tương lai người bệnh có mắc phải bệnh xơ phổi hay không. Để biết tình trạng bệnh, cần phải chụp CT nhiều lần gây tốn kém và có khả năng nhiễm xạ.

Do đó, nhóm đề xuất xây dựng hệ thống dự đoán khả năng mắc bệnh xơ phổi của bệnh nhân dựa vào dữ liệu cá nhân, ảnh chụp CT một lần và FVC cơ sở để dự đoán bệnh một cách chủ động và chính xác hơn.

1.2 Mục tiêu đề tài

Mục tiêu của đề cương luận văn là xây dựng và đánh giá hệ thống dự đoán mức độ suy giảm chức năng phổi để hỗ trợ cho việc chẩn đoán bệnh xơ phổi, đẩy nhanh việc phát hiện lâm sàng, giúp bệnh nhân chủ động theo dõi được tình trạng phổi của mình.

1.3 Phạm vi nghiên cứu

Phạm vi nghiên cứu là tiến hành phân tích ảnh CT phổi của bệnh nhân, xử lí thông tin liên quan đến việc phát triển bệnh xơ phổi. Tập dữ liệu được sử dụng là tập dữ liệu trong cuộc thi do OSIC (Open Source Imaging Consortium)kết hợp với Kaggle tổ chức. Tập dữ liệu bao gồm thông tin cá nhân(giới tính, độ tuổi, tình trạng hút thuốc), ảnh CT,FVC theo từng tuần của từng bệnh nhân. Mô hình sẽ sử dụng là các mô hình đã được công bố như EfficientNet, Multi-Quantile Regression.

1.4 Phương pháp nghiên cứu

Các nội dung được thực hiện trong quá trình thực hiện đề cương luận văn bao gồm:

1. Tìm hiểu về bệnh xơ phổi,đánh giá các yếu tố liên quan đến tiên lượng bệnh và xử lí dữ liệu trong tập dữ liệu có sẵn. Tiền xử lý ảnh CT và thông tin của bệnh nhân.

2. Tìm hiểu về mô hình EfficientNet,Multi-Quantile Regression và một số mô hình liên quan.

3. Tiến hành xây dựng và huấn luyện mô hình dự đoán thông số FVC theo tuần.

4. Đánh giá kết quả mô hình dự đoán, kết luận và viết báo cáo.

Chương 2

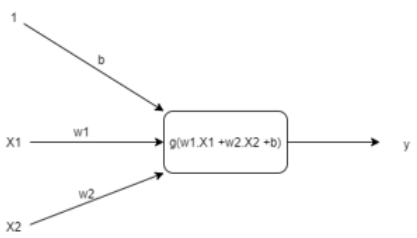
LÝ THUYẾT VỀ CÁC MÔ HÌNH SỬ DỤNG TRONG ĐỀ TÀI

2.1 Lý thuyết mạng nơ-ron

Neural Network là một mạng lưới gồm nhiều lớp được lấy cảm hứng từ neuron ở não người. Ở đó, lớp đầu tiên để đưa các đặc tính của vật cần dự đoán vào được gọi là Input Layer. Và layer cuối cùng mang kết quả dự đoán gọi Output Layer. Một mạng neuron có thể có hoặc không có các lớp ở giữa Input Layer và Output Layer gọi là Hidden Layer, các Hidden Layer này giúp cho tỉ lệ dự đoán chính xác cao hơn tuy nhiên việc huấn luyện cũng tốn nhiều thời gian và dung lượng hơn. Mỗi Layer là tập hợp nhiều node, các node của lớp sau kết nối với toàn bộ các node của lớp trước.

Trong mạng neural network, lớp đầu tiên đưa các đặc tính vào mạng gọi là input layer, lớp cuối cùng để đưa ra các dự đoán gọi là output layer. Giữa 2 lớp input và output gồm một hoặc nhiều lớp ẩn giúp tăng khả năng học và dự đoán của mạng được gọi là hidden layer. Mỗi layer là tập hợp gồm nhiều node, các node của lớp sau kết nối với *toàn bộ* node của lớp trước.

Mỗi node trong hidden layer và output layer thực hiện các công việc sau: Liên kết với tất cả các node ở layer trước đó với các hệ số w riêng. Mỗi node có 1 hệ số bias b riêng. Từ đó w,b biểu thị mối quan hệ giữa node trước và node sau như hình 2.1.



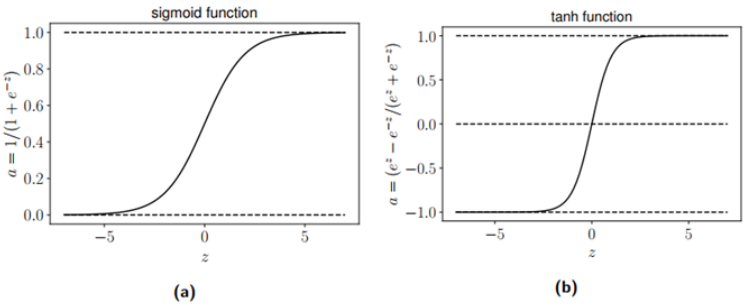
Hình 2.1: Mối quan hệ giữa ngõ vào và ngõ ra của một node trong mang nơ-ron.

Node phía trên có đầu vào là X1,X2, trọng số w1,w2. Ngõ ra y là kết quả

của một hàm phi tuyến và một hàm tuyến tính. Hàm phi tuyến còn được gọi là hàm kích hoạt (activation).Các hàm phi tuyến tính (kích hoạt) thường được sử dụng là:

sigmoid:Hàm sigmoid có công thức: $sigmoid(z) = \frac{1}{1+e^{-z)}}$ với đồ thị như trong hình 2.2(a). Nếu đầu vào lớn, hàm số sẽ cho đầu ra gần với 1. Với đầu vào nhỏ (rất âm), hàm số sẽ cho đầu ra gần với 0.

Tanh: Giá trị ngõ ra được chuyển về trong khoảng $[-1, 1]$ khiến nó có tính chất tâm không (zero-centered)thay vì chỉ dương như sigmoid, có công thức $Tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ với đồ thị như hình 2.2(b).



Hình 2.2: Đồ thị hàm sigmoid (a) và hàm tanh (b) [5].

ReLU và Leaky RELU: RELU lấy ngưỡng giá trị ở 0 (Thay thế các giá trị âm bằng 0): $RELU(z) = max(0, z)$. Từ đây có thể suy ra chức năng của lớp ReLU là chuyển toàn bộ giá trị âm của ngõ vào (là kết quả lấy từ lớp Convolutional) thành giá trị 0, tạo ra tính phi tuyến cho mô hình để phù hợp với các mẫu trong thực tế không phải lúc nào cũng tuyến tính. Lớp ReLU làm biến đổi ngõ vào và xuất thẳng đến ngõ ra do đó không có trọng số được học trong lớp ReLU [5].

2.2 Mang nơ-ron tích chập

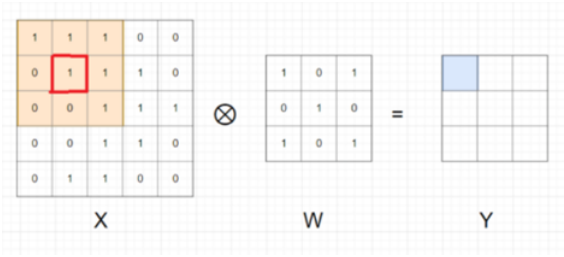
Mạng neural cơ bản khi giải quyết bài toán bài toán xử lý phân loại hình ảnh gặp một số khó khăn. Ví dụ đối với ảnh màu $64 * 64$ được biểu diễn dưới dạng 1 tensor 3 chiều $64 * 64 * 3$. Việc để biểu thị hết nội dung của bức ảnh thì cần truyền vào input layer tất cả các pixel $(64 * 64 * 3 = 12288)$. Nghĩa là input layer giờ có 12288 nodes. Giả sử số lượng node trong hidden layer 1 là 1000. Số lượng weight W giữa input layer và hidden layer 1 là $12288 * 1000 = 12288000$, số lượng bias là 1000 nên tổng số parameter là: 12289000. Đây mới chỉ là số parameter giữa input layer và hidden layer 1, trong model còn nhiều layer nữa, và nếu kích thước cây ảnh tăng, ví dụ $512 * 512$ thì số lượng parameter tăng cực kì nhanh. Điều này khiến cho việc tính toán của máy tính cần rất nhiều công sức nhưng lại không mang lại hiệu quả cao.

Do vậy ý tưởng sử dụng mạng neural tích chập (Convolutional Neural Network -CNN) ra đời. Áp dụng phép tính convolution vào các layer trong Neural Network ta có thể giải quyết được vấn đề giảm thiểu lượng lớn parameter mà vẫn lấy ra được các đặc trưng của ảnh. Một mô hình mạng CNN sẽ có cấu trúc chung gồm các lớp convolution và các lớp khác như ReLu layer, pooling, fully connected.

2.2.1 Phép tính tích chập

Lấy ví dụ trên ảnh xám, ảnh được biểu diễn dưới dạng ma trận A kích thước $m * n$. Ta định nghĩa kernel là một ma trận vuông kích thước $k * k$ trong đó k là số lẻ. k có thể bằng 1, 3, 5, 7, 9,.. Ví dụ kernel kích thước $3 * 3$. Kí hiệu phép tính convolution là $\otimes$, kí hiệu $Y = X \otimes W$.

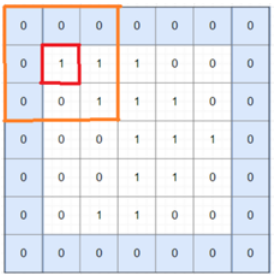
Với mỗi phần tử x_{ij} trong ma trận X lấy ra một ma trận có kích thước bằng kích thước của kernel W có phần tử x_{ij} làm trung tâm gọi là ma trận A. Sau đó tính tổng các phần tử của phép tính element-wise của ma trận A và ma trận W, rồi viết vào ma trận kết quả Y. Hình 2.3 mô tả phép tính tích chập $Y = X \otimes W$.



Hình 2.3: Phép tính tích chập $Y = X \otimes W$.

2.2.2 Phép tính Padding và Stride

Padding: Như ở trên, mỗi lần thực hiện phép tính convolution xong thì kích thước ma trận Y đều nhỏ hơn X. Tuy nhiên giờ ta muốn ma trận Y thu được có kích thước bằng ma trận X, cần hêm giá trị 0 ở viền ngoài ma trận X.

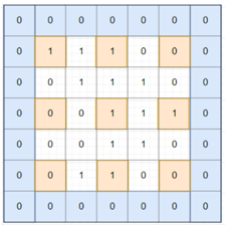


Hình 2.4: Ma trận khi thêm viền 0 bên ngoài.

Hình 2.4 mô tả một ma trận với padding=1 với những ô vuông màu xanh là những ô vuông được thêm vào ma trận ban đầu. Padding = k nghĩa là thêm k vector 0 vào mỗi phía của ma trận.

Stride:Khi thực hiện tuần tự các phần tử trong ma trận X, thu được ma trận Y cùng kích thước ma trận X, ta gọi là $stride = 1$. Tuy nhiên nếu $tride = k(k > 1)s$ thì ta chỉ thực hiện phép tính convolution trên các phần tử $x_{1+i*k,1+j*k}$.

Hình 2.5 là ma trận có padding=1. Stride=2. Hiểu đơn giản là bắt đầu từ vị trí x_{11} sau đó nhảy k bước theo chiều dọc và ngang cho đến hết ma trận X. Kích thước của ma trận Y là $3 * 3$ đã giảm đi đáng kể so với ma trận X. Stride thường dùng để giảm kích thước của ma trận sau phép tính convolution. Công thức tổng quát cho phép tính convolution của ma trận X kích thước m*n với kernel kích thước $k * k$, $stride = s, paddings = p$ ra ma trận Y kích thước :

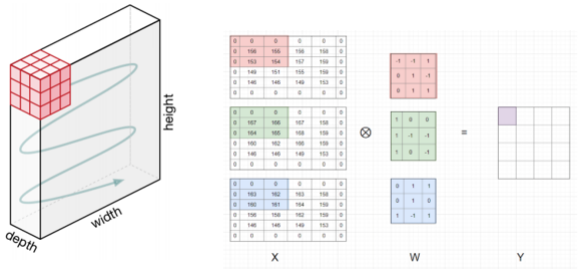


Hình 2.5: Ma trận khi padding = 1, stride = 2.

$$\left(\frac{m-k+2p}{s}+1\right) * \left(\frac{n-k+2p}{s}+1\right) \tag{2.1}$$

2.2.3 Convolutional Layer

Convolutional Layer là khối xây dựng cốt lõi, nơi thực hiện hầu hết những tính toán nâng cao của một mạng neural tích chập. Một lớp Convolutional Layer gômf một bộ các bộ lọc chứa những tham số có khả năng học và cập nhật theo thời gian. Mỗi bộ lọc là một vùng không gian nhỏ, được đại diện bằng một ma trận số thực. Trong suốt quá trình huấn luyện, bộ lọc này sẽ trượt dọc theo chiều dài và chiều rộng của bức ảnh đầu vào, lần lượt quét những vùng không gian con của bức ảnh và trích xuất đặc trưng trong vùng ấy. Những đặc trưng học đượ được lưu trong ma trận gọi là feature map [5].



Hình 2.6: Phép tính tích chập trên ảnh màu với filter 3x3 [5].

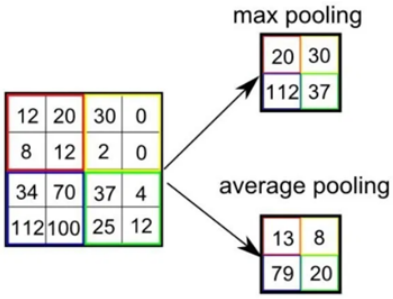
Với mỗi kernel khác nhau ta sẽ học được những đặc trưng khác nhau của ảnh, nên trong mỗi convolutional layer ta sẽ dùng nhiều kernel để học được nhiều thuộc tính của ảnh. Vì mỗi kernel cho ra output là 1 matrix nên k kernel sẽ cho ra k output matrix. Ta kết hợp k output matrix này lại thành 1 tensor 3 chiều có chiều sâu k.

Output của convolutional layer đầu tiên sẽ thành input của convolutional layer tiếp theo. Convolutional layer tổng quát khi giả sử input của 1 convolutional layer tổng quát là tensor kích thước $H * W * D$. Kernel có kích thước $F * F * D$ (kernel luôn có depth bằng depth của input và F là số lẻ), stride: S, padding: P. Convolutional layer áp dụng K kernel. Output của layer là tensor 3 chiều có kích thước:

$$\left(\frac{H-F+2P}{S}+1\right) * \left(\frac{H-F+2P}{S}+1\right) * K \tag{2.2}$$

2.2.4 Pooling Layer

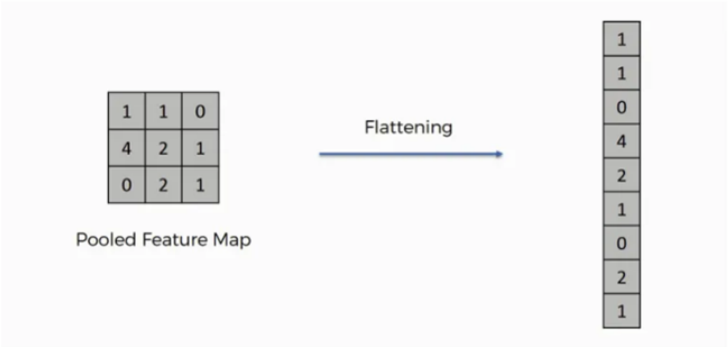
Lớp pooling thường được sử dụng ngay sau lớp convulational để đơn giản hóa thông tin đầu ra để giảm bớt số lượng neuron. Mục đích của pooling rất đơn giản, nó làm giảm số hyperparameter mà ta cần phải tính toán, từ đó giảm thời gian tính toán, tránh overfitting. Có 2 loại pooling layer phổ biến là: max pooling và average pooling.



Hình 2.7: Ví dụ về Pooling Layer [5]

2.2.5 Fully Connected Layer

Sau khi ảnh được truyền qua nhiều convolutional layer và pooling layer thì model đã học được tương đối các đặc điểm của ảnh (ví dụ mắt, mũi, khung mặt,...) thì tensor của output của layer cuối cùng, kích thước $H * W * D$, sẽ được chuyển về 1 vector kích thước $(H*W*D)$. Sau đó ta dùng các fully connected layer để kết hợp các đặc điểm của ảnh để ra được output của model [5].



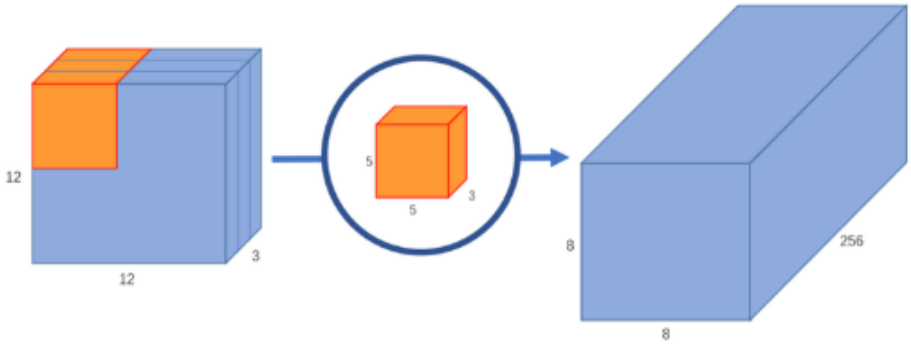
Hình 2.8: Hình phép Flatten biến đổi tensor về 1 vector [5].

2.3 Mạng Mobilenet

Ở mô hình mạng MobileNet v1, ta sử dụng các lớp tích chập depthwise Seperable, một tích chập tối ưu hơn so với tích chập thông thường như đã trình bày ở phần 2.2. Ở mô hinh này, lớp tích chập depthwise sử dụng những filter đơn lẻ cho từng kênh ngõ vào (input channel). Lớp tích chập pointwise sử dụng tích chập 1x1 để kết hợp các ngõ ra của lớp tích chập depthwise. Lớp tích chập

thông thường thực hiện cả việc lọc và kết hợp input trong một lần. Lớp tích chập depthwise separable chia nó ra làm 2 giai đoan, một giai đoạn riêng cho việc lọc và một giai đoạn riêng cho việc kết hợp. Điều này đã giúp mang lại hiệu quả lớn trong việc giảm số lượng tính toán và kích thước mô hình [4].

Giả sử ảnh đầu vào có kích thước $12 * 12 * 3$, thực hiện 1 phép tích chập có cửa số $5 * 5$ lên ảnh đó với $padding = 0$ và $stride = 1$ như hình 2.9. Nếu chỉ xét trên chiều dài và rộng của ảnh thì quá trình chập là $(12 * 12) \rightarrow (5 * 5) \rightarrow (8 * 8)$. Cửa sổ lướt qua toàn bộ ảnh, mỗi lần thực hiện 25 phép nhân và trả về 1 điểm ảnh. Vì $padding = 0$ nên ngõ ra là $(8 * 8)$. Tuy nhiên, ảnh đầu vào có 3 chiều, do đó thay vì thực hiện 25 phép nhân, ta phải thực hiện 75 phép nhân cho mỗi lần kernel dịch chuyển. Giả sử ta sử dụng 256 kernel, vậy ngõ ra sẽ là $8 * 8 * 256$. Để tạo ra được ngõ ra này, ta phải thực hiện $256 * 5 * 5 * 8 * 8 * 3 = 1,228,800$ phép nhân. Đây là một con số vô cùng lớn, do đó mạng nơ ron thông thường không thể sử dụng trong các mô hình nhúng [4].



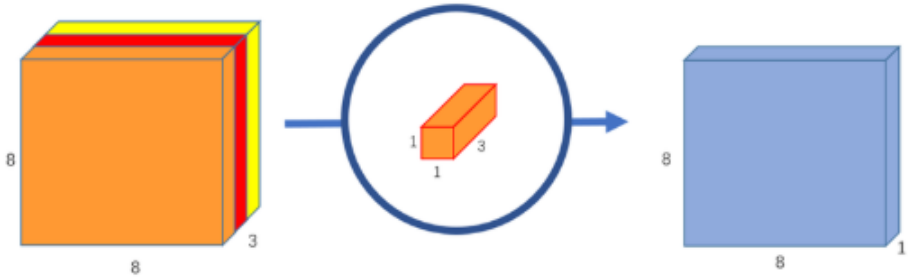
Hình 2.9: Thực hiện tích chập thông thường [4].

Thực hiện phép chập depthwise seperable tức là thực hiện lần lượt phép chập depthwise và phép chập pointwise. Ta thực hiện phép chập depthwise seperable với cùng ví dụ trên với ảnh đầu vào $12 * 12 * 3$. Đầu tiên, thực hiện chập depthwise bằng cách sử dụng 3 kernel $5 * 5 * 1$. Ở đây, mỗi kernel sẽ thực hiện phép chập với mỗi channel của ảnh đầu vào, cho ra output $8 * 8 * 1$, gộp các kết quả lại, ta thu được ảnh đầu ra $8 * 8 * 3$ như hình 2.10.

Hình 2.10: Thực hiện tích chập depthwise [4].

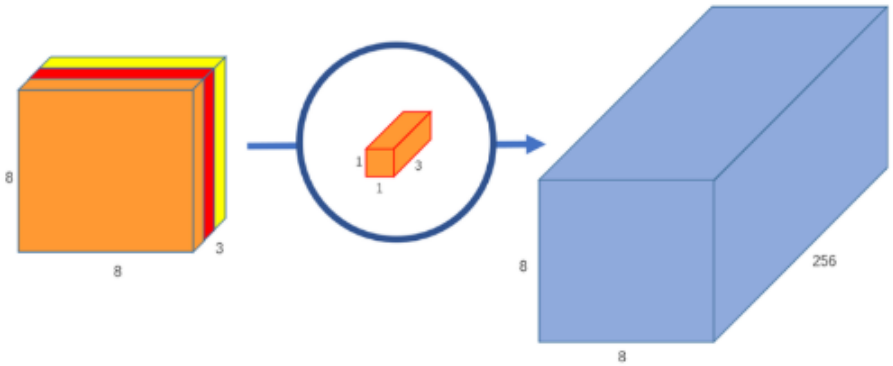
Để thu được ngõ ra có chiều là $8 * 8 * 256$, tiếp theo cần tăng số lượng channel lên 256. Đây là nhiệm vụ của tích chập pointwise. Tích chập này sử dụng các

kernel có kích thước là $1 * 1$ để thực hiện phép chập ở từng điểm dữ liệu, số lượng kernel bằng với số lượng channel của ảnh đâu vào với mục đích thu được 1 channel của ảnh đầu ra. Ở ví dụ này, kernel sẽ là $1 * 1 * 3$, out put sẽ là $8 * 8 * 1$ như hình 2.11.



Hình 2.11: Thực hiện tích chập pointwise,biến đổi ảnh thành 1 channel [4].

Sau đó, sử dụng 256 kernel $1 * 1 * 3$ ta được ảnh đầu ra là $8 * 8 * 256$ như hình 2.12.



Hình 2.12: Sử dụng tích chập pointwise với 256 kernels, ngõ ra là ảnh với 256 kernel [4].

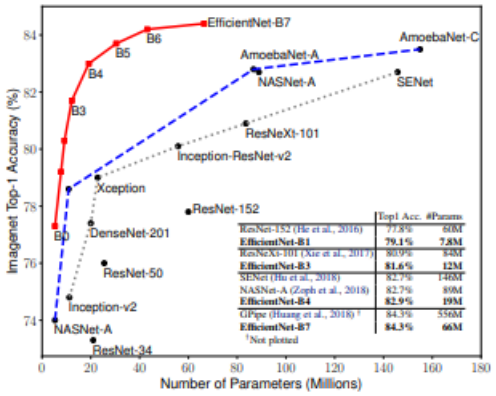
Như vậy, sau khi thực hiện 2 phép chập với cùng một ngõ vào và ngõ ra, ta so sánh số phép nhân cần thực hiện để thấy được đặc điểm nổi bật của tích chập depthwise separable so với tích chập thông thường. Đối với tích chập thông thường, thực hiện chập 256 kernel $5 * 5 * 3$ trên $8 * 8$ điểm của ảnh đầu vào, nghĩa là thực hiện $256 * 3 * 5 * 5 * 8 * 8 = 1,228,800$ phép nhân. Đối với tích chập depthwise separable, khi thực hiện phép chập depthwise, sử dụng 3 kernels $5 * 5 * 1$, kernel dịch chuyển $8 * 8$ lần, nghĩa là thực hiện $3 * 5 * 5 * 8 * 8 = 4,800$ phép nhân. Khi thực hiện chập pointwise, chúng ta có 256 kernel $1 * 1 * 3$ dịch chuyển $8 * 8$ lần, tức là $256 * 1 * 1 * 3 * 8 * 8 = 49,152$ phép nhân. Tổng cộng cả 2 giai đoạn là $53,952$ phép nhân. Số $53,953$ nhỏ hơn $1,228,800$ rất nhiều, do đó mô hình mạng sẽ nhẹ hơn và khả năng thực hiện nhanh hơn rất nhiều lần so với các phép nhân convolution thông thường.

2.4 Mạng EfficientNet

2.4.1 Giới thiệu mạng EfficientNet

Convolutional Neural Networks (ConvNets) thường được hiệu chỉnh một số thông số trong mô hình để tối ưu về độ chính xác và số lượng thông số. Các thông số liên quan đến cấu trúc mạng baseline được hiệu chỉnh gồm độ sâu(depth), độ rộng(width) và độ phân giải(image resolution). Trong mô hình ResNet, tác giả đã điều chỉnh số lượng Layer tạo ra các mô hình như ResNet-18,ResNet-200. Trong mô hình GPipe, tác giả tăng độ lớn của baseline lên 4 lần để đạt được độ chính xác cao hơn. Tuy nhiên, các mô hình này thường điều chỉnh một trong 3 thông số (depth, width, resolution). Ở mô hình EfficientNet, tác giả thay đổi đồng thời cả 3 thông số này một cách có hệ thống dựa theo phương pháp *Compound coefficient*.

Mô hình EfficientNet B7 hiện tại đã đạt "state-of-art" đối với tập ảnh ImageNet khi đạt Top1 accuracy là 84.3% và đối với tập CIFAR-100 là 91.7%. Độ chính xác và số lượng thông số của mạng EfficientNet so với các mạng ConvNet khác được thể hiện trên hình 2.13.



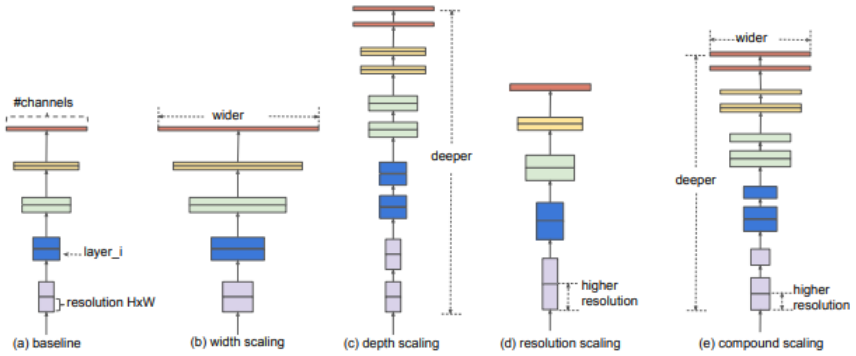
Hình 2.13: So sánh các mô hình trên tập ImageNet [3].

Khi tiếp cận bài toán dự đoán bệnh xơ phổi của bệnh nhân, nhóm quan tâm đến độ chính xác mô hình hơn so với tốc độ xử lý. Vì vậy trong số các mô hình, nhóm nhận thấy EfficientNet (mạng back-bone - trích xuất dữ liệu) của EfficientDet tuy tốc độ xử lí thấp khó đáp ứng realtime nhưng độ chính xác cao nên quyết định sử dụng mạng Efficient Net để dự đoán.

2.4.2 Thu phóng mô hình

Để thay đổi một mô hình sao cho có độ chính xác và tốc độ xử lý cao hơn, các mạng thường tiếp cận theo 4 hướng chính: Thay đổi chiều sâu của mạng, thay đổi chiều rộng của mạng, thay đổi độ phân giải đầu vào và thay đổi cấu trúc của từng lớp. Tuy nhiên mục tiêu mà Efficient Net tiếp cận là sử dụng những mạng cơ bản đơn giản như MobileNet và không thay đổi cấu trúc của mạng này, chỉ thay đổi đồng thời chiều sâu, chiều rộng của mạng cũng như độ phân giải của đầu vào sao cho đạt độ chính xác cao. Điều này được gọi là Model Scaling (thu phóng mô hình).

Thu phóng theo chiều sâu được thực hiện bằng cách thêm hoặc bớt một số lớp cho mô hình. Việc thu phóng mô hình theo chiều sâu nếu hợp lý có thể trích



Hình 2.14: Cách thu phóng mô hình (a) Mô hình gốc (b) Mô hình thu phóng theo chiều rộng (c) Mô hình thu phóng theo chiều sâu (d) Mô hình thu phóng theo độ phân giải (e) Mô hình thu phóng kết hợp 3 loại thu phóng theo chiều sâu, rộng và độ phân giải [3].

xuất được các đặc trưng phức tạp và phong phú hơn. Thu phóng theo chiều rộng tức là thêm dữ liệu đầu vào và thêm các node cho từng lớp. Việc thu phóng mô hình theo chiều rộng cũng cần hợp lý, nếu đủ rộng các đặc trưng được trích xuất một cách chi tiết hơn. Thu phóng theo độ phân giải là việc tăng giảm độ phân giải cho dữ liệu ảnh đầu vào để bức hình có thêm nhiều đặc trưng hơn. Thu phóng độ phân giải một cách hợp lý cũng giúp cho mô hình thu được nhiều đặc trưng có chi tiết nhỏ mà có thể bị bỏ sót nếu không thu phóng độ phân giải. Tuy nhiên nếu 3 việc thu phóng này diễn ra một cách không hợp lý, tăng quá sâu hay quá rộng mô hình, tăng độ phân giải lên quá cao sẽ dễ dẫn tới overfitting hay vanishing gradient do học được quá nhiều đặc trưng mang tính cá biệt.

Như vậy, tác giả gọi $Y_i = \mathcal{F}_i(X_i)$ trong đó Yi là tensor ngõ ra của lớp thứ i, Xi là tensor ngõ vào của lớp thứ i và Fi là toán tử được thực hiện ở lớp thứ i. Vậy với một mạng có k lớp, mạng đó được định nghĩa công thức (2.3):

$$\mathcal{N} = \mathcal{F}_k \odot \mathcal{F}_{k-1} \odot ... \odot \mathcal{F}_1 = \odot_{j=1..k} \mathcal{F}_j(X_1) \tag{2.3}$$

Nhiều lớp có cấu trúc giống nhau thường được gom lại thành 1 tầng (1 stage). Khi đó công thức (2.3) được viết lại như công thức (2.4).

$$\mathcal{N} = \odot_{i=1...s} \mathcal{F}_i^{L_i} X_{<H_i, W_i, C_i>} \tag{2.4}$$

Trong đó i là chỉ số tầng, H_i, W_i, C_i lần lượt là chiều dài, chiều rộng và chiều sâu của đầu vào X.Hàm mục tiêu của EfficientNet là

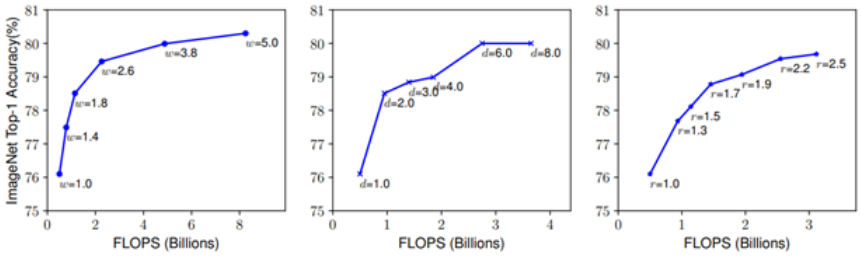
$$MaxAccuracy(\odot_{i=1..s} \hat{F}_i^{d.\hat{L}_i} X_{<r.\hat{H}_i, r.\hat{W}_i, w.\hat{C}_i>}$$

Trong đó r,w,d là thông số hiệu chỉnh mạng, $\hat{F}_i, \hat{L}_i, \hat{W}_i, \hat{H}_i, \hat{C}_i$ là thông số của mạng baseline trước khi hiệu chỉnh.

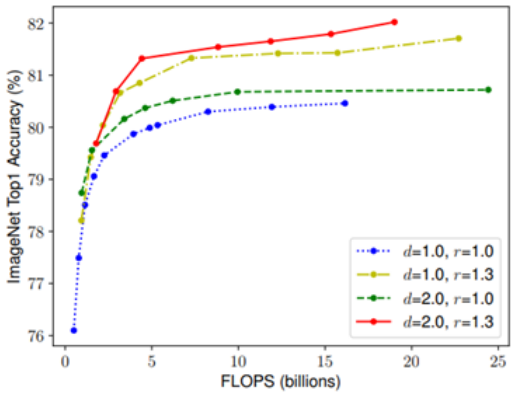
2.4.3 Thu phóng kết hợp

Nhóm tác giả thực hiện thu phóng trên tập ImageNet với các kích thước khác nhau và nhận thấy rằng các chiều của mô hình không thu phóng độc lập với nhau. Nếu thực hiện thu phóng độc lập như hình 2.15, độ chính xác của mạng

ban đầu tăng nhanh nhưng sau đó trở nên bão hòa và không thay đổi. Nhưng nếu thực hiện thu phóng kết hợp, độ chính xác của mô hình tăng rõ rệt như hình 2.16.



Hình 2.15: Thu phóng lần lượt theo độ rộng, độ sâu và độ phân giải, ta thấy với ban đầu FLOPS (tốc độ tính số floating point trên giây) và độ chính xác ban đầu tăng rõ rệt nhưng càng về sau càng bão hòa [3].



Hình 2.16: Thực hiện thu phóng đồng thời cả chiều rộng và chiều sâu (màu đỏ) cho kết quả cải thiện [3].

Từ đó tác giả đề cập đến một hệ số ϕ là mối liên hệ giữa chiều dài, chiều rộng và chiều sâu:

$$d = \alpha^{\phi}$$

$$w = \beta^{\phi}$$

$$r = \gamma^{\phi}$$

$$\alpha.\beta^2.\gamma^2 \approx 2$$

$$\alpha \geq 1, \beta \geq 1, \gamma \geq 1$$

Trong đó ϕ là hệ số do có thể chỉ định để kiểm soát chiều dài, chiều sâu và độ phân giải của mô hình, α, β, γ là chỉ cách gán cho chiều dài, chiều sâu và độ phân giải. Kèm theo đó tác giả cũng có ràng buộc . Hệ số FLOPS tỉ lệ thuận với chiều sâu, bình phương chiều rộng và bình phương độ phân giải. Vì vậy tác giả ràng buộc để FLOPS sẽ tăng theo tỉ lệ 2^{ϕ}

2.4.4 EfficientNet B0 - EfficientNet B7

Tương tự mạng MnasNet, tác giả cũng sử dụng hàm tối ưu mục tiêu cho mô hình là:

$$ACC(m) * \left[\frac{FLOPS(m)}{T}\right]^w$$

Trong đó T là mức FLOPS mục tiêu, $ACC(m)$ và $FLOPS(m)$ là độ chính xác và FLOPS mục tiêu của mô hình m, $w = -0.07$ là siêu tham số để kiểm soát sự cân bằng giữa ACC và FLOPS. Mục tiêu FLOPS của EfficientNet là vào khoảng 400M.Đầu tiên, tác giả sử dụng mô hình Efficient Net B0 đơn giản như hình 2.17:

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

Hình 2.17: Mô hình Efficient Net B0 [3].

Trong đó MBConv là mô hình Mobile Convolution Network. Từ mô hình Base 0 này, đầu tiên tác giả giữ nguyên giá trị $\phi = 1$. Rồi áp dụng gmall grid tìm kiếm và ràng buộc $\alpha.\beta^2.\gamma^2 \approx 2$ để tính ra 3 giá trị còn lại $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$. Tiếp tục, giữ cố định α, β, γ rồi thu phóng mạng với ϕ khác nhau để thu được các EfficientNet từ B0 đến B7. Kết quả thực nghiệm so sánh trên hình 2.18 cho thấy kết quả tốt hơn so với các mô hình trước.

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPs	Ratio-to-EfficientNet
EfficientNet-B0	**77.1%**	**93.3%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**79.1%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**80.1%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.6%**	**95.7%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.9%**	**96.4%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.6%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.8%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.3%**	**97.0%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

Hình 2.18: So sánh độ chính xác top-1 (kết quả có dự đoán cao nhất trùng với kết quả), độ chính xác top-5(trong 5 kết quả dự đoán cao nhất có kết quả), số tham số,FLOPS [3].

2.5 Mô hình Linear Regression

Regression là phương pháp nghiên cứu mối quan hệ giữa 2 biến mà cụ thể một biến sẽ là biến độc lập, biến còn lại sẽ là biến mục tiêu (bị ảnh hưởng bởi biến độc lập) , mô hình hóa, định lượng hóa mối quan hệ này để qua đó có thể xác định được giá trị của biến mục tiêu nếu các biến độc lập thay đổi như thế nào.

Linear regression là mô hình nghiên cứu mối quan hệ tuyến tính giữa một số biến độc lập và biến phụ thuộc. Phương trình tổng quát là

$$y \approx f(x) = \hat{y}$$

$$f(x) = w_1x_1 + w_2x_2 + w_3x_3 + w0$$

Với w_1, w_2, w_3, w_0 là các hằng số. Bài toán đi tìm các hệ số tối ưu w_1, w_2, w_3, w_0.

Đặt $\mathbf{w} = [w_0, w_1, w_2, w_3]^T$ là vector cột cần phải tối ưu và $\overline{\mathbf{x}} = [1, x_1, x_2, x_3]$ là vector hàng dữ liệu đầu vào mở rộng, khi đó, phương trình tôngr quát được viết lại là :

$$y \approx \overline{\mathbf{x}}\mathbf{w} = \hat{y}$$

Hàm mất mát của bài toán Linear Regression là

$$\mathcal{L}(\mathbf{w} = \frac{1}{2}\sum_{i=1}^{N}(y_i - \overline{\mathbf{x}}\mathbf{w})^2$$

Trong lúc huấn luyện, ta mong muốn sự mất mát nhỏ nhất, ta đạo hàm theo $\mathbf{w}$ của hàm mất mát:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = \overline{\mathbf{X}}^T(\overline{\mathbf{X}}\mathbf{w} - \mathbf{y})$$

Với khái niệm giả nghịch đảo, điểm tối ưu của bài toán Linear Regression có dạng:

$$\mathbf{w} = \mathbf{A} \dagger \mathbf{b} = (\overline{\mathbf{X}}^T)\overline{\mathbf{X}} \dagger \overline{\mathbf{X}}^T\mathbf{y}$$

Quantile Regression là mô hình hổi quy mở rộng của hồi quy tuyến tính - Linear regression, tìm hiểu mối quan hệ tuyến tính giữa biến độc lập và biến phụ thuộc trong trường hợp bộ dữ liệu có các giá trị ngoại lệ (outliers), độ lệch/ chiều cao của phân phối dữ liệu (high skewness), mức độ không đồng nhất của dữ liệu.

Mô hình dựa trên xem xét phân phối tổng thể của dữ liệu, không chỉ sử dụng mỗi giá trị trung bình để tính toán, xây dựng công thức như linear regression.

Quantile hay còn gọi là phân vị trong thống kê, là phương pháp xác định với $n\%$ bất kì của bộ dữ liệu thì phân phối các giá trị của dữ liệu trong $n\%$ là như thế nào (như các giá trị đã được sắp xếp từ nhỏ tới lớn) để đánh giá độ phân tán của dữ liệu, và tại phân vị thứ n này giá trị của biến là bao nhiêu.

$$y = \beta' X_i + \varepsilon_i \tag{2.5}$$

Công thức tính sai số có trọng số theo mô hình hồi quy:

$$\tau \sum_{y_i > \hat{\beta}'_\tau X_i} |y_i - \hat{\beta}'_\tau X_i| + (1-\tau) \sum_{y_i < \hat{\beta}'_\tau X_i} |y_i - \hat{\beta}'_\tau X_i| \tag{2.6}$$

Phương trình tổng quát của Quantile Regression tương tự như Linear regression, tuy nhiên Quantile Regression hướng đến giảm thiểu sai số của mô hình với phương trình như phương trình (2.6) .

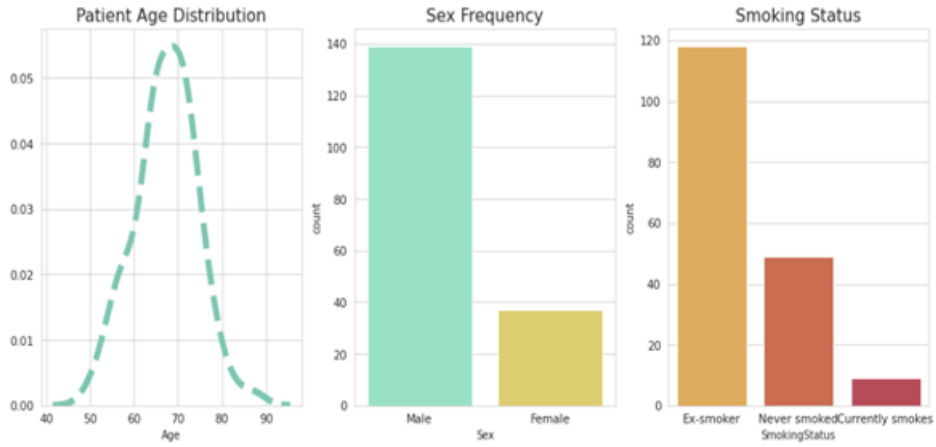
Chương 3

GIẢI PHÁP THỰC HIỆN MÔ HÌNH DỰ ĐOÁN BỆNH XƠ PHỔI

3.1 Xây dựng và tiền xử lý tập dữ liệu

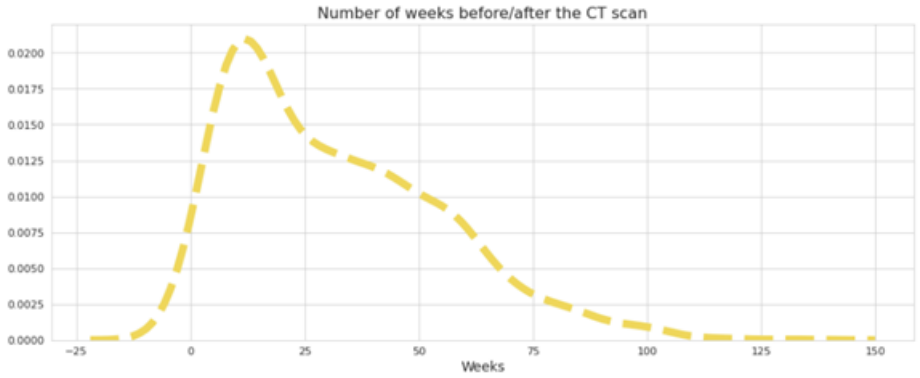
3.1.1 Tập dữ liệu

Tập dữ liệu gồm 33202 hình ảnh CT của 176 bệnh nhân và các siêu dữ liệu về chỉ số FVC theo một số tuần được đo, giới tính, tuổi tác và tình trạng hút thuốc. 176 bệnh nhân có độ tuổi từ 50 đến 90 trong đó phần đông các bệnh nhân có tuổi từ 60 đến 70. Về giới tính, đa phần là bệnh nhân nam. Về tình trạng hút thuốc là những người đã từng hút thuốc. Biểu đồ trong hình 3.1 thể hiện dữ liệu về bệnh nhân.



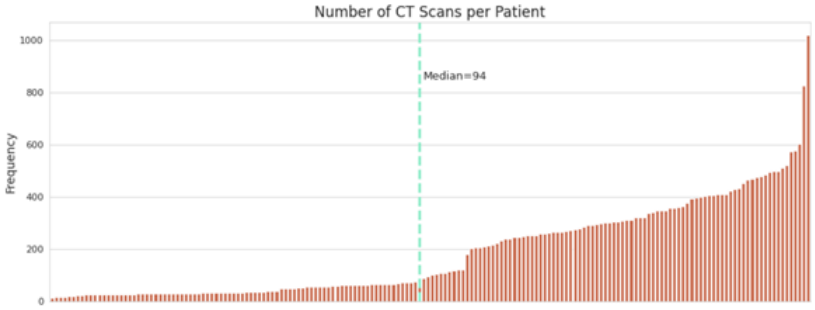
Hình 3.1: Biểu đồ thống kê thông tin của bệnh nhân trong tập dữ liệu

FVC (Forced Vital Capacity) hay dung tích sống gắng sức là lượng không khí thở ra nhanh và mạnh sau khi gắng sức hít thở sâu nhất có thể. FVC của các bệnh nhân sẽ được đo trong nhiều tuần trước, sau và ngay trong tuần chụp CT như hình 3.2. Tùy từng bệnh nhân mà thời gian đo có thể khác nhau, trải dài từ 5 tuần trước khi hình CT được chụp (-5) đến 133 tuần sau khi chụp CT (133).

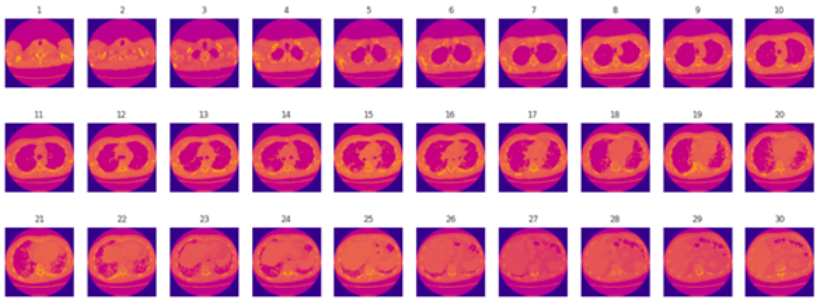


Hình 3.2: Thời gian các ảnh CT được chụp

Về ảnh chụp CT các bệnh nhân, số lượng ảnh chụp CT các bệnh nhân cũng khác nhau. Ảnh chụp CT chụp liên tục phổi bệnh nhân trong một lần hít vào và khi quá trình hết một vòng tuần hoàn hô hấp hít vào như thế máy quét CT cũng dừng lại. Do đó, tùy vào tình trạng phổi mà bệnh nhân có thể hít sâu hay không khi đó số lượng ảnh chụp CT cũng khác nhau. Hình 3.4 thể hiện ảnh chụp CT của một bệnh nhân. Hơn một nửa số lượng bệnh nhân có số lượng ảnh ít hơn 100 trong đó trung bình là 94 ảnh. Số lượng ảnh chụp CT của bệnh nhân được thể hiện ở hình 3.3.



Hình 3.3: Số lượng ảnh chụp CT của từng bệnh nhân



Hình 3.4: Ví dụ ảnh chụp CT trong một lần hít vào của 1 bệnh nhân

3.1.2 Tiền xử lí ảnh

Ảnh CT chụp thông tin về mật độ bức xạ của một thể hoặc các phần mô mềm khi tiếp xúc với tia X. Một lát cắt ngang của hình CT được tái cấu trúc sau khi các phép đo của tia X được thực hiện theo nhiều hướng khác nhau. Do vậy tùy thuộc vào các thành phần quang phổ do cài đặt thông số đo lường của các máy chụp CT mà mức xám của các pixel thể hiện phổi của bệnh nhân có thể khác nhau, hình ảnh phổi của bệnh nhân cũng to hay nhỏ tùy thuộc vào các thiết lập này.

3.1.2.1 Biến đổi HU và thay đổi kích thước ảnh

Đơn vị HU (Hounsfield Unit) ra đời để chuẩn hóa các mức xám của hình chụp CT. Và từ đơn vị HU này người ta cũng có thể xác định các thành phần có trong ảnh chụp CT như không khí, phổi, mỡ,... theo hình 3.5. Ảnh được chuyển qua đơn vị HU như hình 3.6
Để đổi đơn vị HU, ta có thể áp dụng công thức sau: $U = m * SV + SI$.

Substance	HU
Air	−1000
Lung	−500
Fat	−100 to −50
Water	0
CSF	15
Kidney	30
Blood	+30 to +45
Muscle	+10 to +40
Grey matter	+37 to +45
White matter	+20 to +30
Liver	+40 to +60
Soft Tissue, Contrast	+100 to +300
Bone	+700 (cancellous bone) to +3000 (cortical bone)

Hình 3.5: Đơn vị HU tương ứng với các thành phần trong ảnh chụp CT

Trong đó:

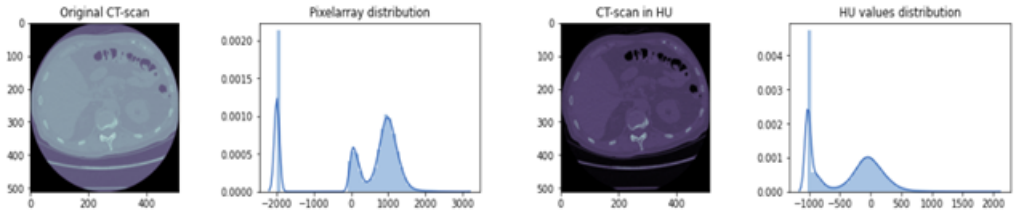
m là giá trị bit đầu vào của ảnh chụp CT trước khi đổi sang HU.

SV là giá trị Rescale Slope, một trong các tham số được lưu trong file dicom với tag là $(0028, 1052)$.

SI là giá trị Rescale Intercept, một trong các tham số được lưu trong file dicom với tag là $(0028, 1053)$

U là giá trị HU đầu ra.

Ảnh sau khi được biến đổi HU để đồng nhất đơn vị cho các pixel, ta thực hiện thay đổi cắt và resize toàn bộ các ảnh về kích thước $512x512$. Bằng cách đổi toàn bộ các đơn vị ma trận về dạng int32 rồi thực hiện biến đổi sang hình ảnh nhờ thư viện Pillow. Từ hình ảnh sau khi được biến đổi, thực hiện cắt ảnh, đưa trở về dạng array nhờ thư viện Numpy và lưu lại ở dạng .npy.

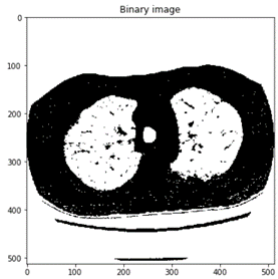


Hình 3.6: Ảnh chụp CT trước và sau khi chuyển đổi sang đơn vị HU

3.1.2.2 Phân đoạn ảnh chụp CT

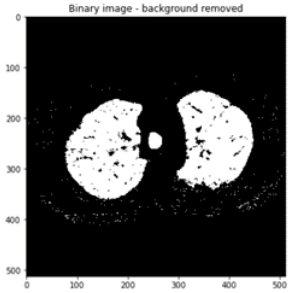
Sau khi đồng nhất các ảnh về kích thước $512 * 512$, dựa vào đơn vị HU ta có thể dễ dàng phân đoạn ảnh đối với vị trí phổi. Để phân đoạn phổi đầu tiên, thiết lập ngưỡng -320 đối với các pixel để xử lý sang ảnh nhị phân. Các pixel có giá trị thấp hơn -320 ta sẽ đưa về 0 còn những pixel lớn hơn -320 ta sẽ đưa về 1 (Hình 3.7). Khi này ta sẽ phân đoạn được 2 vùng:

- Vùng có giá trị 0(trắng): Vùng không khí và phổi.
- Vùng có giá trị 1(đen): Các vật thể khác.



Hình 3.7: Ảnh chụp CT sau khi được phân ngưỡng -320 để tách làm 2 nhóm

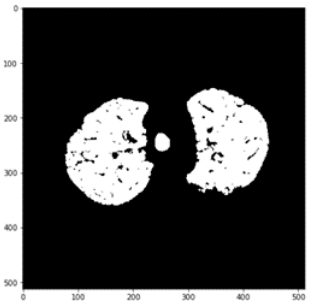
Như vậy, ta chỉ cần phân đoạn được không khí và phổi với nhau thì sẽ thu được kết quả cuối cùng sau khi phân đoạn. Để thực hiện phân đoạn giữa phổi và không khí ta thực hiện phân đoạn bằng cách gán nhãn với các pixel có cùng giá trị ở cạnh nhau. Những pixel mang giá trị HU là không khí tập trung ở viền bức ảnh, do vậy với các pixel có cùng nhãn ta sẽ tăng thêm 1 để loại bỏ nó. Sau khi thực hiện, ta sẽ chỉ còn phần phổi và các đốm trắng nhiễu như hình 3.8:



Hình 3.8: Ảnh chụp CT sau khi loại bỏ phần không khí

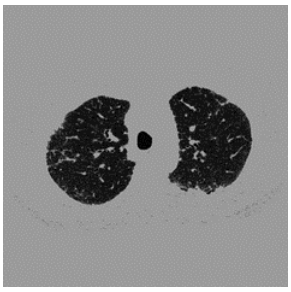
Kế đó, để loại bỏ nhiễu ta sẽ áp dụng hình thái đóng (closing morphology). Closing morphology thực hiện 2 phép toán giãn nở rồi xói màn. Với phép giãn

nở, các bit có giá trị 1 sẽ được giãn nở với filter là một hình dĩa có bán kính là 2. Sau khi thực hiện phép giãn nở các chấm nhiễu sẽ biến mất và ta thực hiện phép xói màn với filter có cùng kích thước để đưa hình về ban đầu (Hình 3.9).



Hình 3.9: Ảnh chụp CT sau khi được phân đoạn

Kết quả cuối cùng sẽ trở thành mask để lọc các giá trị pixel nằm ở vị trí của phổi trong ảnh được lưu ở dạng .npy ở phần resize ảnh. Cuối cùng ta thu được ảnh đã được lọc vị trí của phổi và lưu nó ở dạng .png bằng mode = "gray" của thư viện matplolib (Hình 3.10).



Hình 3.10: Ảnh chụp CT đã được lọc vị trí của phổi

3.2 Xây dựng mô hình dự đoán FVC

Để thực hiện dự đoán, mô hình được xây dựng gồm 2 mô hình con là Efficient Net và Quantile Regression. Dữ liệu hình ảnh chụp CT và các dữ liệu của bệnh nhân sẽ được đưa vào Efficient Net, lúc này Efficient Net có nhiệm vụ dự đoán hệ số góc của phương trình giữa FVC và tuần của bệnh nhân rồi từ đó suy ra FVC. Do phương trình đường thẳng giữa FVC và tuần là đường thẳng gần đúng mà không thể hiện được chính xác được mối quan hệ giữa FVC và tuần, khi đó các FVC đã dự đoán được ở mô hình Efficient Net (EN) sẽ tiếp tục được đưa và mô hình hồi quy Quantile Regression (QR) một lần nữa cùng với các dữ liệu của bệnh nhân để từ đó dự đoán FVC. FVC cuối cùng sẽ được tính theo công thức sau:

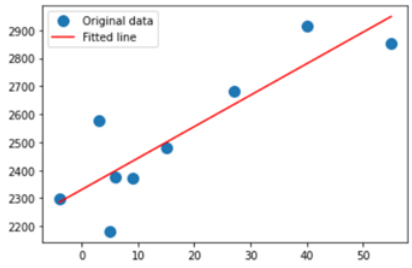
$$FVC = FVC(EN) * 0.55 + FVC(QR) * 0.45 \tag{3.1}$$

3.2.1 Huấn luyện mô hình Efficient Net

3.2.1.1 Chuẩn bị dữ liệu

Ở mô hình Efficient Net ta thực hiện dự đoán dựa trên các hình chụp CT. Tuy nhiên do tập dữ liệu quá lớn nên RAM của Google Colab không thể xử lý toàn bộ ảnh của bệnh nhân được. Trong tập dữ liệu ảnh chụp CT của các bệnh nhân thông thường sẽ gồm 3 giai đoạn: bắt đầu hít sâu, thực hiện hít sâu và bắt đầu thở ra. Do đó để giải quyết về vấn đề RAM của Google Colab, mỗi bệnh nhân nhóm chỉ chọn ra ngẫu nhiên 1 hình trong từng giai đoạn đó. Như vậy với dữ liệu đầu vào là ảnh chụp CT ta có 3 hình x 176 bệnh nhân. Vậy kích thước ngõ vào là $(528, 64, 64)$.

Các dữ liệu khác về độ tuổi, giới tính và tình trạng hút thuốc cũng ảnh hưởng đến FVC, vì vậy các dữ liệu cũng sẽ được mã hóa để đưa vào mô hình. Đối với độ tuổi, ta sẽ chuẩn hóa dữ liệu về quanh mức 1. Đối với giới tính, 0 sẽ là nam, 1 sẽ là nữ. Đối với tình trạng hút thuốc, "00" là chưa từng hút thuốc, "11" là đã từng hút thuốc, "01" là hiện đang hút thuốc. Với từng bệnh nhân, từ các giá trị FVC và tuần, tìm ra đường thẳng xấp xỉ mối quan hệ giữa FVC và tuần rồi từ đó lưu lại hệ số góc của đường thẳng này (a) làm mục tiêu cho việc huấn luyện mô hình Efficient Net.



Hình 3.11: Phương trình đường thẳng (màu đỏ) quan hệ giữa tuần (trục hoành) và FVC(trục tung) và FVC chính xác theo từng tuần (chấm màu xanh) của bệnh nhân "ID00076637202199015035026"

3.2.1.2 Mô hình huấn luyện Efficient Net

Cài đặt thông số:
- Epoch: 30
- Batch size: 8
- Learning rate: 0.03
- Số bước trong một Epoch: 32 (tập huấn luyện) và 16 (tập đánh giá)
- Mô hình tối ưu theo dạng Adam Optimization.

Mô hình được thư viện Keras cài đặt sẵn, như vậy chỉ cần xài mô hình đã được Keras cài sẵn để huấn luyện cho hình chụp CT. Ảnh chụp CT sau khi đi qua Efficient Net được lấy Pooling như sau:

Lớp	Đầu vào	Kích thước đầu vào	Đầu ra	Kích thước đầu ra
Efficient Net	Ảnh chụp CT	$(64, 64, 1)$	X1	
GlobalPooling2D	X1		X2	$(1, 1, 1)$
Concatenate	[X2,dữ liệu]	$(1, 1, 1) * (4, 1, 1)$	X3	(5,1,1)
Drop out	X3	(5,1,1)	X4	(5,1,1)
Fully Connected	X4	(5,1,1)	a	(1,1,1)

Sau khi quá trình huấn luyện kết thúc, ta sẽ có được file các hệ số của mô hình được lưu ở dạng file .h5.

3.2.1.3 Dự đoán FVC từ mô hình Efficient Net

Toàn bộ hình ảnh bệnh nhân trong tập kiểm tra sau khi tiền xử lý sẽ cùng với các thông số của bệnh nhân đi qua mô hình Efficient Net đã được huấn luyện để dự đoán hệ số góc của phương trình đường thẳng giữa FVC và tuần (a). Vì ở tập kiểm tra này ta đưa toàn bộ hình và với mỗi hình lại dự đoán ra được một a cho bệnh nhân. Vì vậy, với mỗi bệnh nhân được dự đoán rất nhiều a (do mỗi bệnh nhân lại có rất nhiều hình chụp CT) xác suất a dự đoán có dạng xác phân phối Gauss, và a cuối cùng đại diện cho kết quả dự đoán của một bệnh nhân sẽ là giá trị ở đỉnh của xác suất Gauss này (tức là giá trị Q2 ở hình 3.12.

Hình 3.12: Phân phối Gauss

Từ a sau khi dự đoán, FVC của 1 tuần được đo để tìm ra giá trị b của phương trình đường thẳng $FVC = a * Tun + b$. Từ phương trình đường thẳng vừa được xác lập tìm FVC từng tuần theo đường thẳng này. Khoảng sai lệch có thể tin tưởng (Confidence) được tính bằng công thức sau:

$$Confidence = Percent - A * |Week[test] - week| \tag{3.2}$$

Trong đó :

Percent là một trường tính toán xấp xỉ phần trăm độ giữa FVC của bệnh nhân với FVC của những người có tình trạng tương tự.

A là hệ số góc của phương trình đường thẳng vừa tìm được.

Week[test] là một số tuần được cho trong tập test.

week là tuần cần dự đoán.

3.2.2 Huấn luyện mô hình Quantile Regression

Các FVC vừa tìm được ở mô hình Efficient Net là các FVC dựa vào phương trình đường thẳng mà tìm được do vậy các FVC ở đây vẫn không chính xác, cần xây dựng thêm mô hình hồi quy tuyến tính để dự đoán.

3.2.2.1 Chuẩn bị dữ liệu

Ở bước này, dữ liệu ở tập train được sử dụng để huấn luyện, dữ liệu tập test để đánh giá và tập sub để kiểm tra.Các dữ liệu được đưa vào mô hình gồm:

- Male: "1" nếu bệnh nhân mang giới tính là nam
- Female: "1" nếu bệnh nhân mang giới tính là nữ
- Ex-smoker : "1" nếu bệnh nhân đã từng hút thuốc
- Never smoked : "1" nếu bệnh nhân chưa từng hút thuốc
- Currently smokes: "1" nếu bệnh nhân vẫn đang hút thuốc
- Age: Chuẩn hóa tuổi bệnh nhân về trong khoảng [0:1]
- Week: Chuẩn hóagiá trị FVC của các bệnh nhân về từ [0:1]
- Base: Điều chỉnh lại tuần cần dự đoán với tuần đầu tiên cần xác định là tuần 0

Các thông số của mô hình với tối ưu Adam (Adam Optimizer) được thiết lập như sau: Learning rate: 0.1; $\beta_1 : 0.9$; $\beta_2 : 0.999$; epoch: 600; batch size:8.

3.2.2.2 Xây dựng mô hình

Mô hình hồi quy tuyến tính Quantile Regression được mô tả như bảng sau:

Lớp	Đầu vào	Kích thước đầu vào	Đầu ra	Kích thước đầu ra
Fully Connected 1	Dữ liệu bệnh nhân	$(767, 8)$	d1	$(767, 100)$
Fully Connected 2	d1	$(767, 100)$	d2	$(767, 100)$
Fully Connected 3	d2	$(767, 100)$	p1	$(767, 3)$
Fully Connected 4	d2	$(767, 100)$	p2	$(767, 3)$
Lambda	[p1,p2]	[(767,3)(767,3)]	preds	(767,3)

Ở lớp cuối Lambda không thực hiện các tích chập mà được tính như sau:

$$pred = p1 + cumsum(p2)$$

Trong đó:

pred là ma trận dự đoán đầu ra ở lớp cuối

p1 là ma trận đầu vào thứ nhất

cumsum(p2) là tổng cộng dồn các giá trị của ma trận đầu vào thứ 2 qua 767 bệnh nhân

Ma trận đầu ra thực hiện dự đoán 3 giá trị là 3 giá trị FVC với mức lượng tử lần lượt là 0.2, 0.5, 0.8 tức là 3 giá trị Q1, Q2, Q3 theo hình 3.12 . Trong đó mức lượng tử 0.5 là giá trị FVC dự đoán, FVC trong khoảng từ 0.2 đến 0.8 là mức sai số tin tưởng được (Confidence) của mô hình. Hàm mất mát (Loss function) của mô hình được tính như sau:

$$L = 0.8 * Mean(max(q * (y_{true} - y_{prediction}), (q-1) * (y_{true} - y_{prediction}))) + (1 - 0.8) * score$$

Trong đó:

L là giá trị mất mát

q là các ngưỡng lượng tử [0.2 0.5 0.8]

y_{true} là giá trị kết quả chính xác

$y_{prediction}$ là giá trị kết quả do mô hình dự đoán
score là hàm đánh giá dựa trên kỹ thuật đánh giá Laplace Log Likehood.

Laplace Log Likelihood là phương pháp đánh giá được sử dụng trong y học để đánh giá độ chính xác của các mô hình. Mô hình được xây dựng để phản ánh cả độ sự chính xác và mức độ tin tưởng trên từng dự đoán. Mô hình cụ thể như sau:

$$\sigma = y_{prediction}[q = 0.8] - y_{prediction}[q = 0.2]$$

$$FVC_{prediction} = y_{prediction}[q = 0.5]$$

$$\sigma_{clipped} = max(\sigma, 70)$$

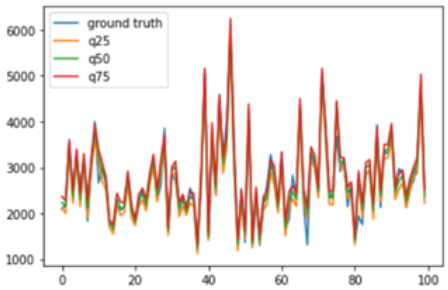
$$\Delta = min(|FVC_{true} - FVC_{prediction}|, 1000)$$

$$score = -\frac{\sqrt{2}\Delta}{\sigma_{clipped}} - ln(\sqrt{2}\sigma_{c}lipped)$$

Để tránh việc sai số dự đoán quá lớn ảnh hưởng đến dự đoán, ngưỡng 1000 được thiết lập. Mức thấp nhất của khoảng sai số tin tưởng được được thiết lập ở ngưỡng 70.

3.2.2.3 Dự đoán FVC của mô hình Quantile Regression và kết quả dự đoán FVC cuối cùng

FVC cuối cùng của mô hình được dự đoán từ giá trị lượng tử q = 0.5 của mô hình. Trong khi đó, giá trị tin tưởng (Confidence) là hiệu của giá trị FVC với mới lượng tử q = 0.8 và giá trị FVC với mức lượng tử q = 0.2.



Hình 3.13: : Giá trị FVC với 3 mức lượng tử q = 0.2 (màu cam), q =0.5 (màu xanh lá), q = 0.75 (màu đỏ) và FVC chính xác (màu xanh) của tập đánh giá

Chương 4

KẾT QUẢ HIỆN TẠI ĐẠT ĐƯỢC

4.1 Kết quả đánh giá trên tập Test

FVC và Confidence tìm được khi sử dụng mô hình Efficient Net được lưu vào một file .csv như hình 4.1 tạm gọi là tập sub để đánh giá tổng hợp.

	Patient_Week	FVC	Confidence
0	ID00419637202311204720264_-12	3305.668049	355.854904
1	ID00421637202311550012437_-12	3167.362113	510.407405
2	ID00422637202311677017371_-12	2215.668169	362.340662
3	ID00423637202312137826377_-12	3754.207844	539.466747
4	ID00426637202313170790466_-12	3113.707872	260.532840

Hình 4.1: FVC và Confidence dự đoán được từ mô hình với 5 bệnh nhân ở tuần -12

Các dữ liệu của bệnh nhân đưa vào mô hình Quantile Regression để dự đoán ra FVC(QR).Hình 4.2 là kết quả dự đoán qua mô hình Quantile Regression của bệnh nhân có ID là "ID00419637202311204720264".

	Patient_Week	FVC	Confidence	FVC1	Confidence1
1540	ID00419637202311204720264_-12	3020	355.854904	3039.885010	231.893066
1541	ID00419637202311204720264_-11	3020	339.984457	3035.962158	233.722168
1542	ID00419637202311204720264_-10	3020	324.114010	3032.039185	235.551270
1543	ID00419637202311204720264_-9	3020	308.243562	3028.116089	237.380249
1544	ID00419637202311204720264_-8	3020	292.373115	3024.193237	239.209595

Hình 4.2: FVC dự đoán của mô hình với bệnh nhân "ID00419637202311204720264" với 5 tuần từ tuần -12 tới tuần -8, FVC1 và Confidence 1 là kết quả dự đoán của mô hình

Hai kết quả trên được tính toán theo công thức đã nêu $FVC = FVC(EF) * 0.45 + FVC(QR) * 0.55$ để tính toán ra FVC cuối cùng:

	Patient_Week	FVC	Confidence
0	ID00419637202311204720264_-1	3057.194606	220.183513
1	ID00419637202311204720264_-10	3140.888771	275.404503
2	ID00419637202311204720264_-11	3150.188108	281.540198
3	ID00419637202311204720264_-12	3159.487377	287.675893
4	ID00419637202311204720264_-2	3066.493943	226.319141

Hình 4.3: FVC và Confidence dự đoán cuối cùng từ 2 mô hình ở một số tuần của bệnh nhân "ID00419637202311204720264"

Thực hiện dự đoán trên 5 bệnh nhân với mỗi bệnh nhân gồm 9 giá trị FVC ở các tuần khác nhau ở tập kiểm tra, kết quả của mô hình cho thấy trung bình mỗi bệnh nhân cho thấy có khoảng từ 0 đến 2 giá trị FVC của bệnh nhân bị tính sai trong khoảng từ Confidence. Như vậy, khả năng dự đoán chính xác của mô hình vào khoảng 76.9231%. Các giá trị dự đoán sai khỏi khoảng Confidence cũng không quá cao, không vượt quá 300.

Ngoài ra độ lệch trung bình giữa FVC đúng so với FVC chính xác (lượng tử là 0.5) là 126.8966 Mô hình có tỉ lệ dự đoán chính xác khác cao, tuy nhiên vẫn còn hạn chế. Việc vẫn còn hạn chế nằm ở tập các giá FVC của các bệnh nhân chưa đủ nhiều, việc xây dựng mô hình hồi quy tuyến tính còn hạn chế và do hạn chế ở RAM tính toán nên việc phải resize ảnh chụp CT từ 512 x 512 về còn 64 x 64 khiến mô hình dự đoán thiếu chính xác.

Tài liệu tham khảo

[1] British Lung Foundation. Idiopathic pulmonary fibrosis statistics, 2016.

[2] Pumonary Fibrosis Foundation. Pulmonary Fibrosis Overview, 2019.

[3] Mingxing Tan, Quoc V.Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *CornellUniversity*, 2020.

[4] Dat Nguyen. Separable Convolutions - Toward realtime object detection applications, 2019.

[5] Nguyễn Thanh Tuấn. *Deep Learning cơ bản*. 2019.

[6] Vũ Hữu Tiệp. *Machine Learning cơ bản*. Nhà Xuất Bản Khoa học & kỹ thuật, 2020.